Gingl Zoltán, 2020, Szeged

Mikrovezérlők Alkalmazástechnikája

Digitális perifériák és használatuk

Flash memória

Flash memória

- Tartalmát megőrzi tápfeszültség nélkül is
- Ez a programmemória, konstansok tárolása
- Néhány mikrovezérlőn: scratchpad memory
  - Adatok tárolására speciális hely
- Endurance – hányszor írható (≈10k-100k)
- Data retention – az adatokat meddig őrzi (≈20-100 év)
- Biztonsági bitek
  - titkosítás lehetséges: olvasása tiltható (de törölhető marad)

Flash memória ▸ Írás

- Program letöltése: debug adaterrel (JTAG/C2)
- Programból: MOVX utasítással is írható
- **Írás: csak 0 írható, 1 nem.**
- **Törlés: az összes bit 1-re állítása**
- Gyártói ajánlás:
  - A tápfeszültség-monitort kötelező bekapcsolni és RESET forrásként konfigurálni!
  - Biztonsági okokból van erre szükség

Flash memória ▸ Írás

- Engedélyező SFR bitek
  - **PSWE** (program store write enable)
  - **PSEE** (program store erase enable)
- Írás: PSWE=1
- Törlés: PSWE=1 és PSEE=1
- Az íráshoz/törléshez még kulcs is kell:
  - FLKEY=0xA5;
  - FLKEY=0xF1;
  - Minden egyes írás és törlés előtt szükséges

Flash memória ▸ Írás

- F410: 512-byte méretű lapokból áll
  - Törölni csak teljes lapot lehet
  - Írni bájtonként lehet
- Ha egy lapon levő bájto(ka)t módosítani szeretnénk:
  - a lap használt bájtjainak átmeneti mentése (XRAM)
  - a lap törlése
  - a használt bájtok szükség szerinti módosítása
  - A használt bájtok visszaírása a flash memóriába

Flash memória ▸ Lap törlése – példakód

```
void FLASH_PageErase (unsigned int addr)
{
  bit ea_save = EA;                   // Preserve EA
  char xdata * data pwrite;           // FLASH write pointer

  EA = 0;                             // Disable interrupts
  VDM0CN = 0xA0;                      // Enable VDD monitor
  RSTSRC = 0x02;                      // VDD mon reset source
  pwrite = (char xdata *) addr;       // set pointer
  FLKEY  = 0xA5;                      // Key Sequence 1
  FLKEY  = 0xF1;                      // Key Sequence 2
  PSCTL |= 0x03;                      // PSWE = 1; PSEE = 1
  VDM0CN = 0xA0;                      // enable VDD monitor
  RSTSRC = 0x02;                      // VDD mon resetsource
  *pwrite = 0;                        // Initiate page erase
  PSCTL &= ~0x03;                     // PSWE = 0; PSEE = 0
  EA = ea_save;                       // Restore interrupts
}
```

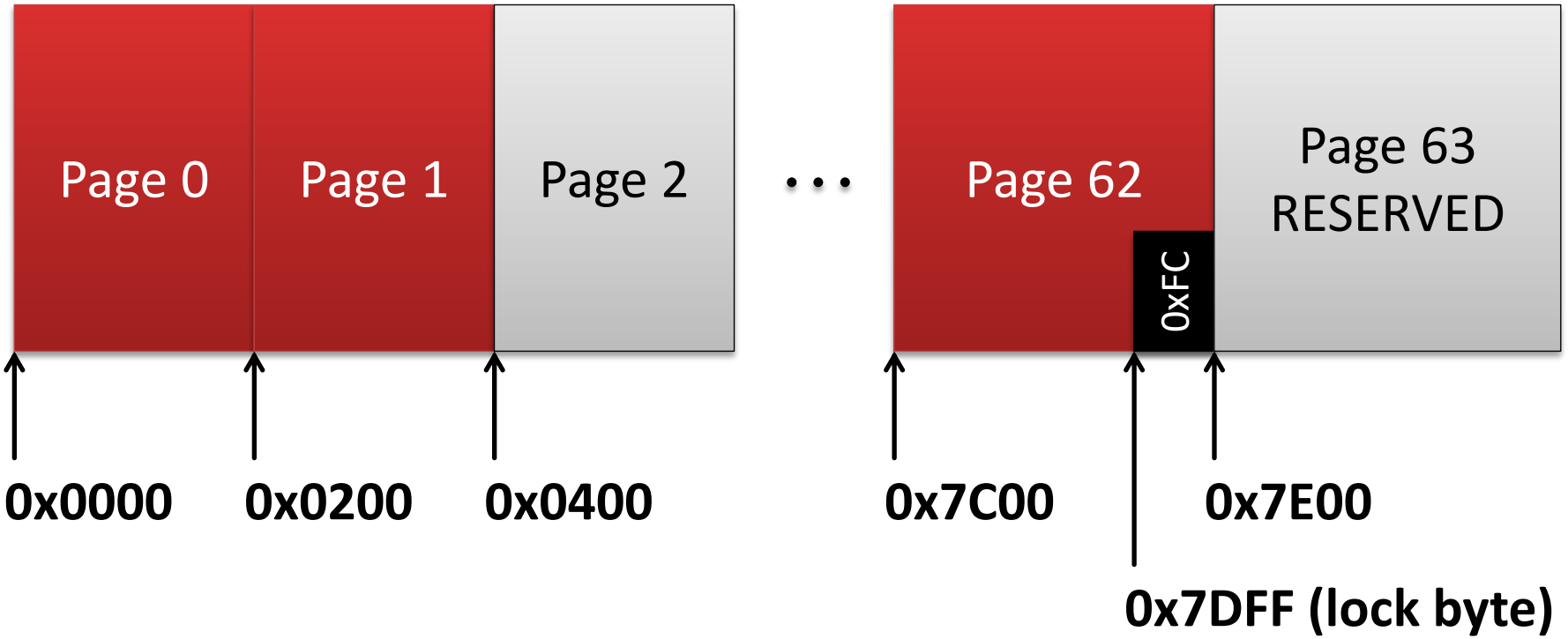
Flash memória ▸ Írás – példakód

```
void FLASH_ByteWrite (unsigned int addr, char byte)
{
  bit ea_save = EA;                   // Preserve EA
  char xdata * data pwrite;           // FLASH write pointer

  EA = 0;                             // Disable interrupts
  VDM0CN = 0xA0;                      // Enable VDD monitor
  RSTSRC = 0x02;                      // VDD monitor reset
  pwrite = (char xdata *) addr;       // set pointer
  FLKEY  = 0xA5;                      // Key Sequence 1
  FLKEY  = 0xF1;                      // Key Sequence 2
  PSCTL |= 0x01;                      // PSWE=1, enable write
  VDM0CN = 0xA0;                      // Enable VDD monitor
  RSTSRC = 0x02;                      // VDD monitor reset
  *pwrite = byte;                     // Write the byte
  PSCTL &= ~0x01;                     // PSWE=0,disable write
  EA = ea_save;                       // Restore interrupts
}
```

Flash memória ▸ Lock byte (F410)

- Lock: védelem kiolvasás ellen
- lock byte: 0x7DFF flash címen érhető el
- n = lock byte értéke negálva:
  - ennyi blokk zárolva (első n-1 blokk és a lock byte blokk)
- Az alábbi példa: 0xFC negálva: 0x03



Flash memória ▸ Hasznos információk

- VDD monitort RESET után azonnal engedélyezni kell RESET forrásként
- Írás előtt a lap használt tartalmát el kell menteni, majd a lapot törölni kell
- Íráskor, törléskor belső órajelre célszerű kapcsolni
- Írás (byte): ~50 us
- Lap törlés (512 byte): ~20 ms
- Bájt olvasás: 40ns
  - max: 25MHz
  - Nagyobb sebesség: prefecth (pipeline)

PORT Input/Output
GPIO (general purpose I/O)

Port I/O ▸ mihez szükséges?

- **Logikai jellel vezérelhető eszközökhöz, írásra:**
  - Áramkörök logikai bemeneteinek meghajtása
  - LED-ek vezérlése, külső elemek vezérlése
  - Külső meghajtó-fokozattal:
    - relé (jelfogó), mágnesszelep, motorok, stb.
- **Logikai jeleket adó eszközökhöz, olvasásra:**
  - Kapcsoló, nyomógomb, billentyűzet
  - Áramkörök logikai kimeneteinek figyelése
  - Külső elektronikával:
    - fénykapuk, Hall-detektorok, stb.

Általános célú port input/output – GPIO

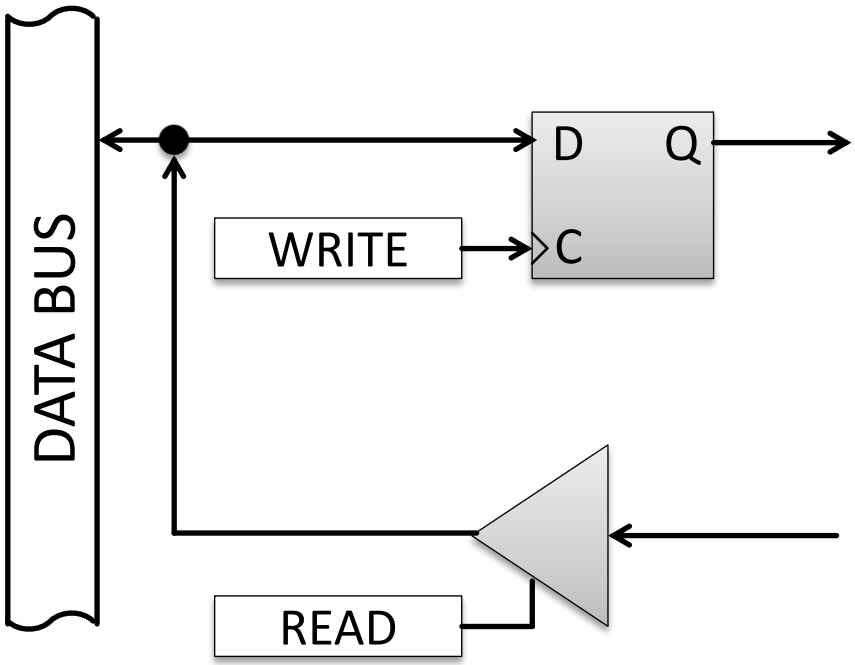
- **A cél összefoglalása:**
  - **Egy logikai jel a külvilág számára**
  - **Külső logikai jel értékének beolvasása**
- Szükséges periféria, mert a processzorok adatbuszán csak rövid ideig érvényes az adat
  - Számos periféria azonos buszt használ
  - Kétirányú busz
  - A busz szélessége adott, korlátozott

A portok 8 bites szervezésűek, bitenként elérhetők

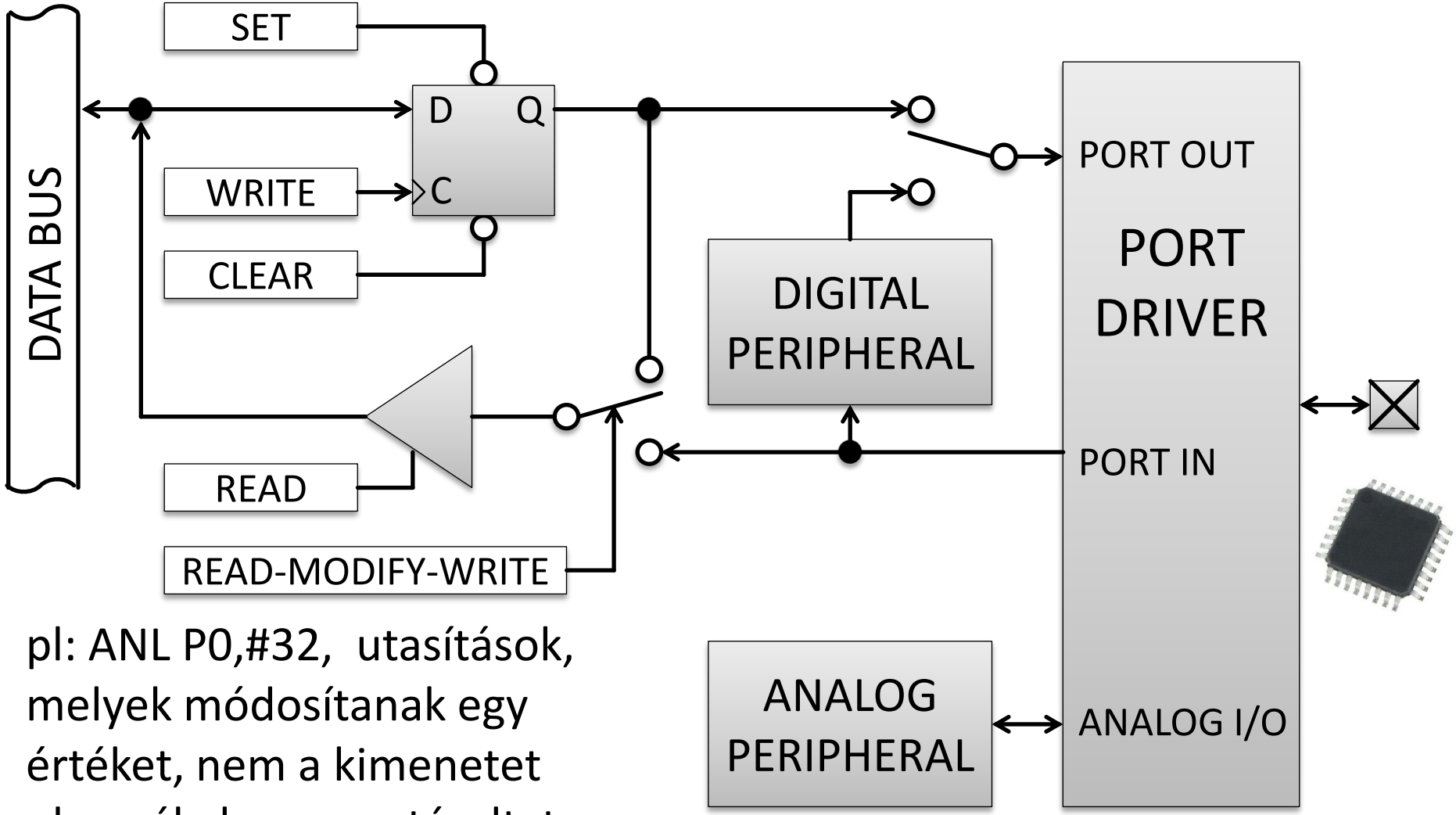
- Alap 8051-en:
  - 4 port: P0, P1, P2, P3
- 8051 verziók:
  - lehet több is vagy kevesebb
- Mindegyik porthoz:
  - 8 áramköri kivezetés
- A kivezetések angol neve:
  - pin
- Elérés SFR regiszterként:
  - direkt címzéssel
- Assembly példák:
  - mov P0, #32
  - mov portvalue,P2
  - setb P1.4
- C példák:
  - P0=32;
  - portvalue=P2;
  - P0_4=1;

Port I/O ▸ megoldás

- **Kimenet**: D tároló (latch)
  - a beírt adat (Q) a következő írásig azonos marad
- **Bemenet**: 3 állapotú meghajtó
  - olvasáskor az adatbuszra tudja kapcsolni a jelet



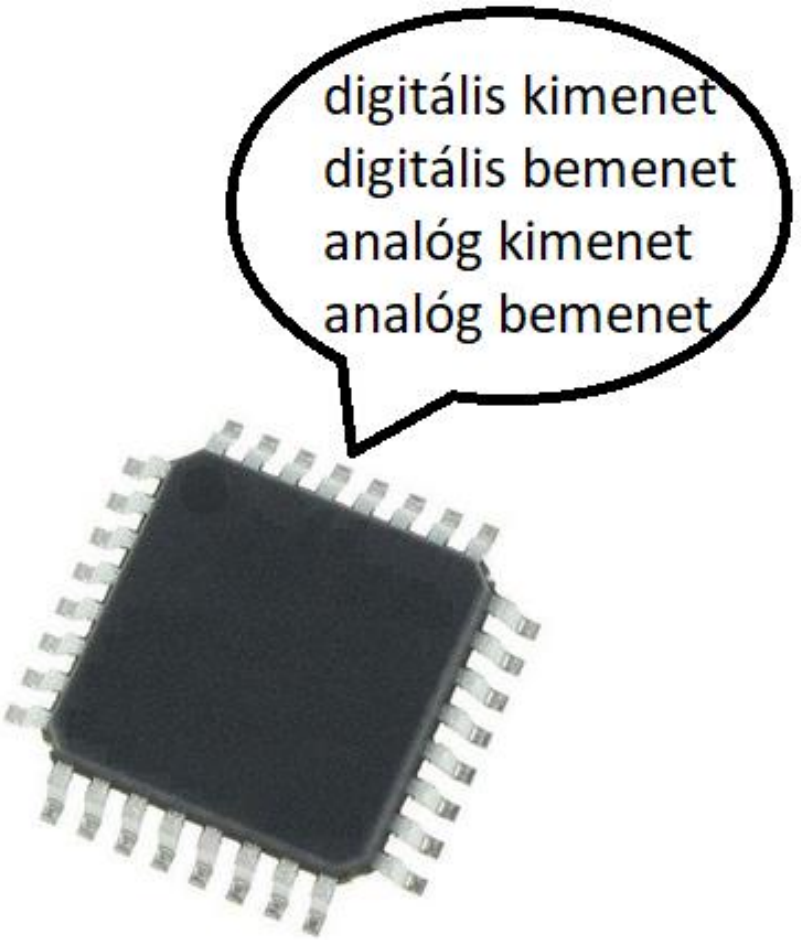
Port I/O ▸ teljesebb kép



pl: ANL P0,#32, utasítások, melyek módosítanak egy értéket, nem a kimenetet olvassák, hanem a tároltat

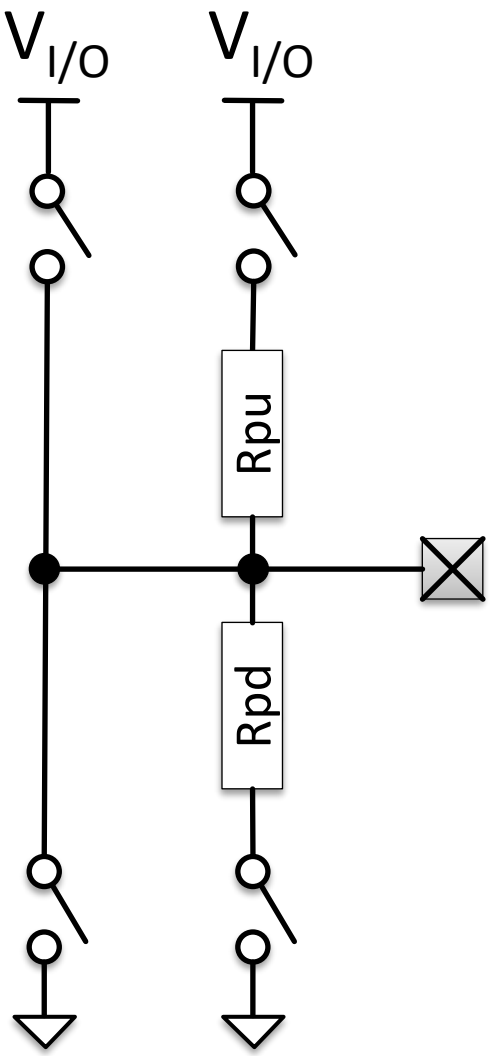
Port I/O ▸ a meghajtó rész felépítése

- Sokféle mód egyetlen kivezetés esetén
- A digitális ki- és bemenetek többfélék is lehetnek

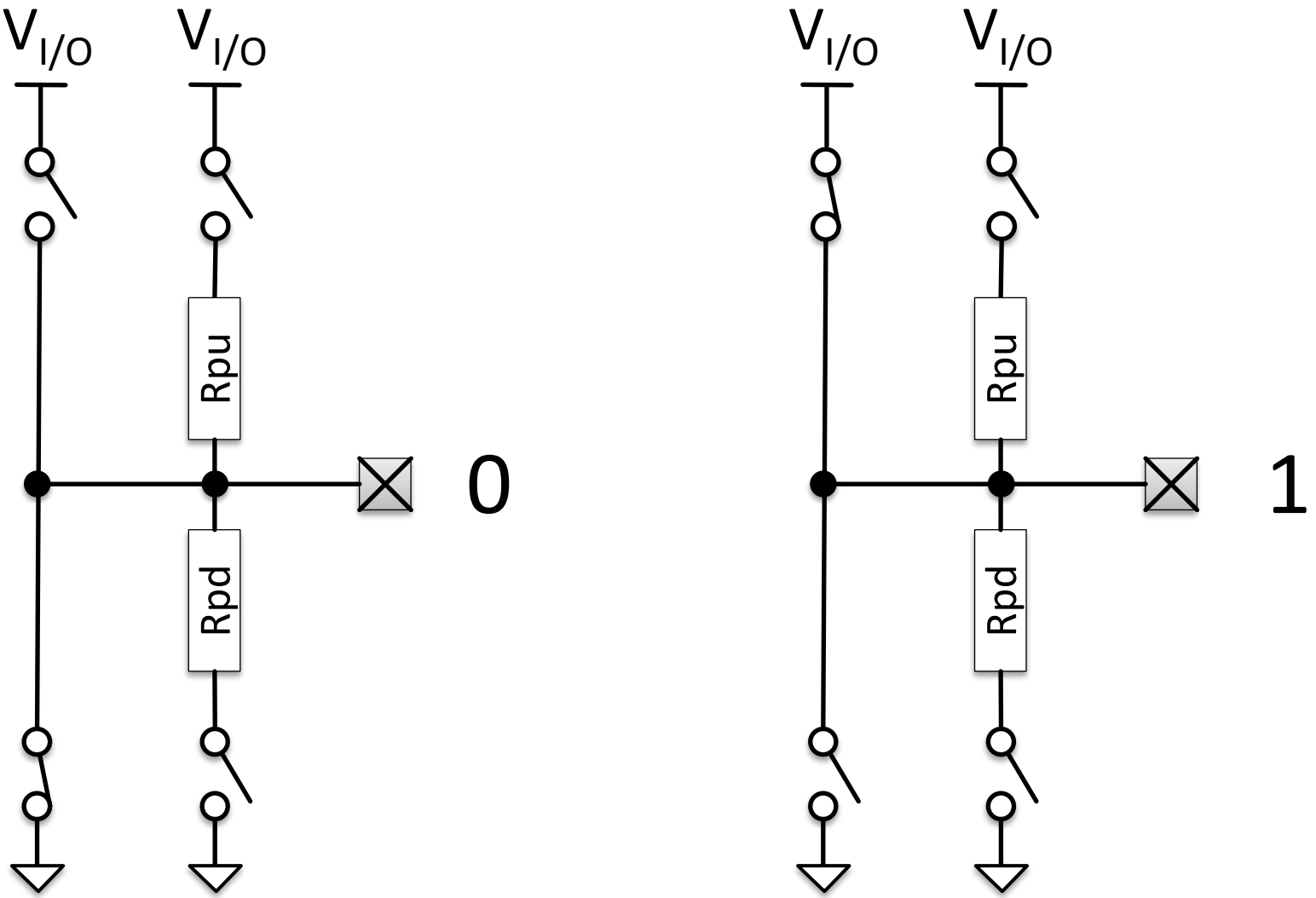


Port I/O ▸ a meghajtó rész felépítése

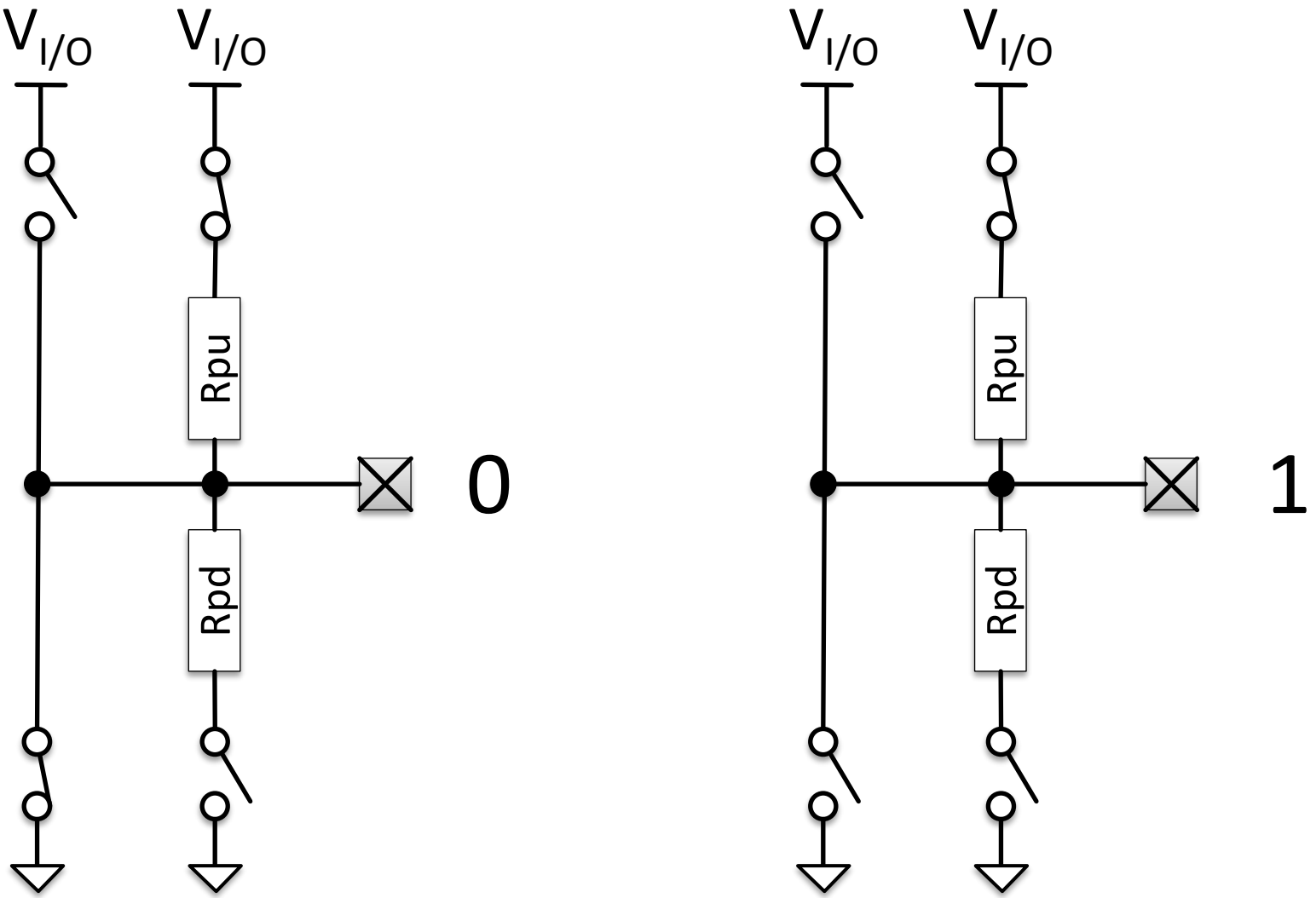
- A kapcsolók (FET-ek) szabják meg
  - A kiválasztott módot
  - A logikai értéket
- A kapcsolók tranzisztorok
  - Digitális jelek vezérlik őket
- $V_{I/O}$: input/output tápfeszültség
  - Általában nagyobb a mikrovezérlő belső áramköreinek tápfeszültségénél (V_{dd})



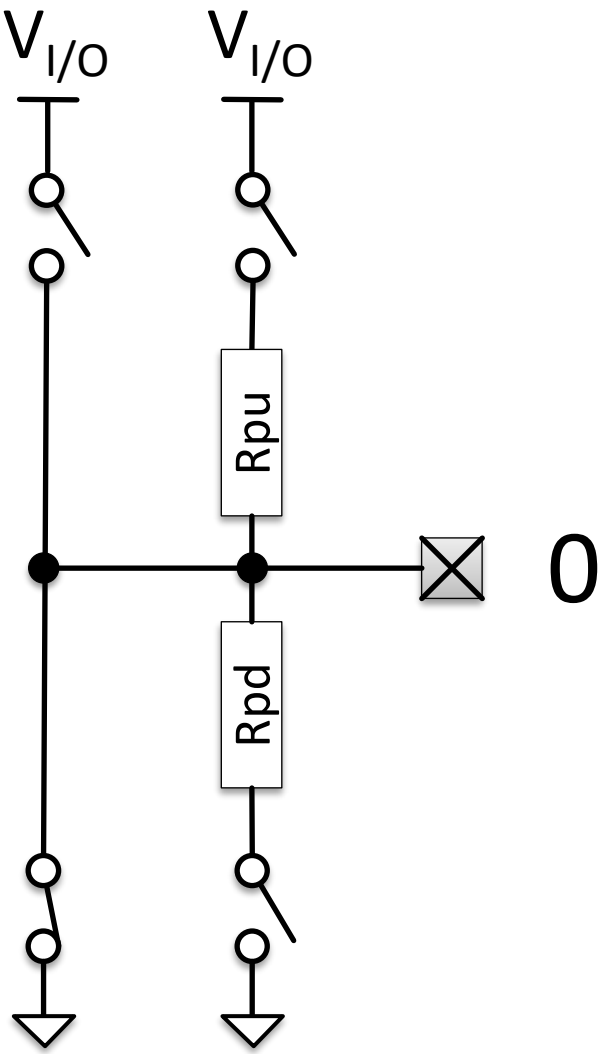
Port I/O ▸ Digital output: push-pull



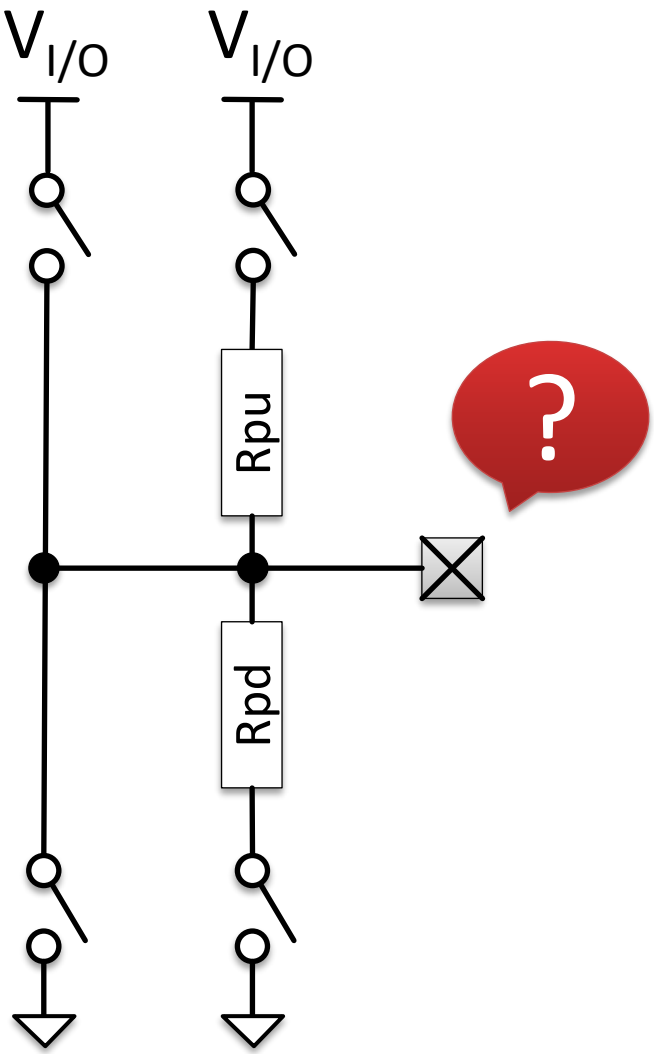
Port I/O ▸ Digital output: open drain with pull-up



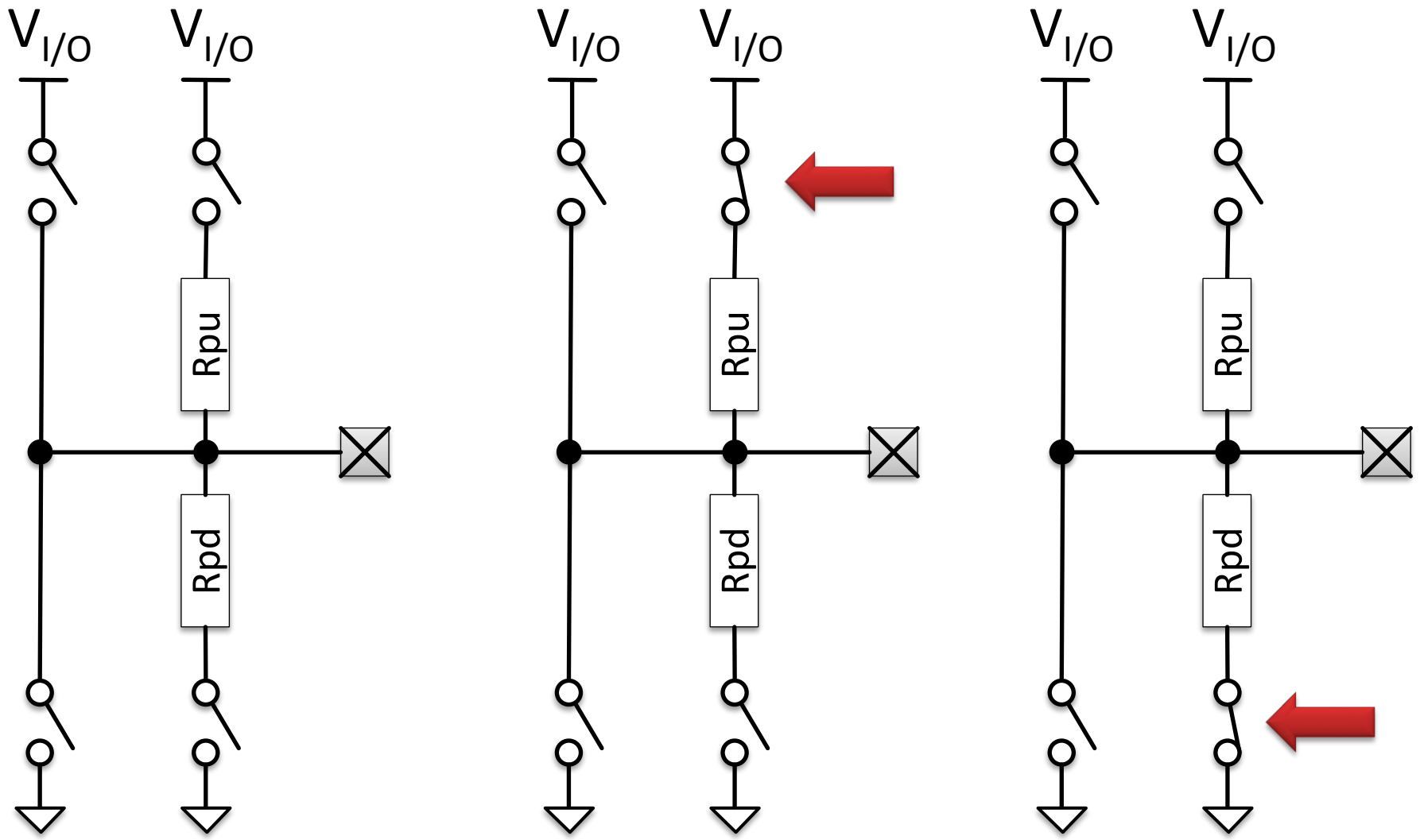
Port I/O ▸ Digital output: open drain



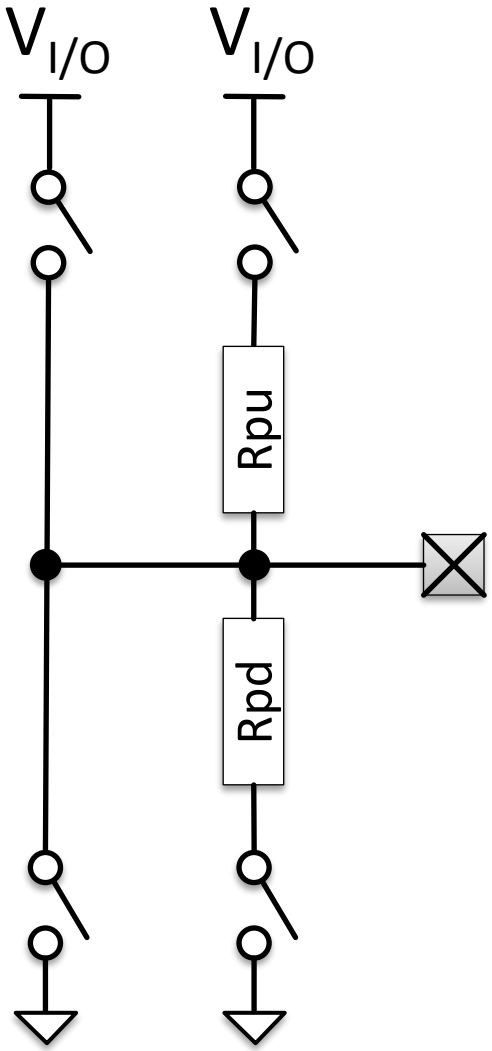
0



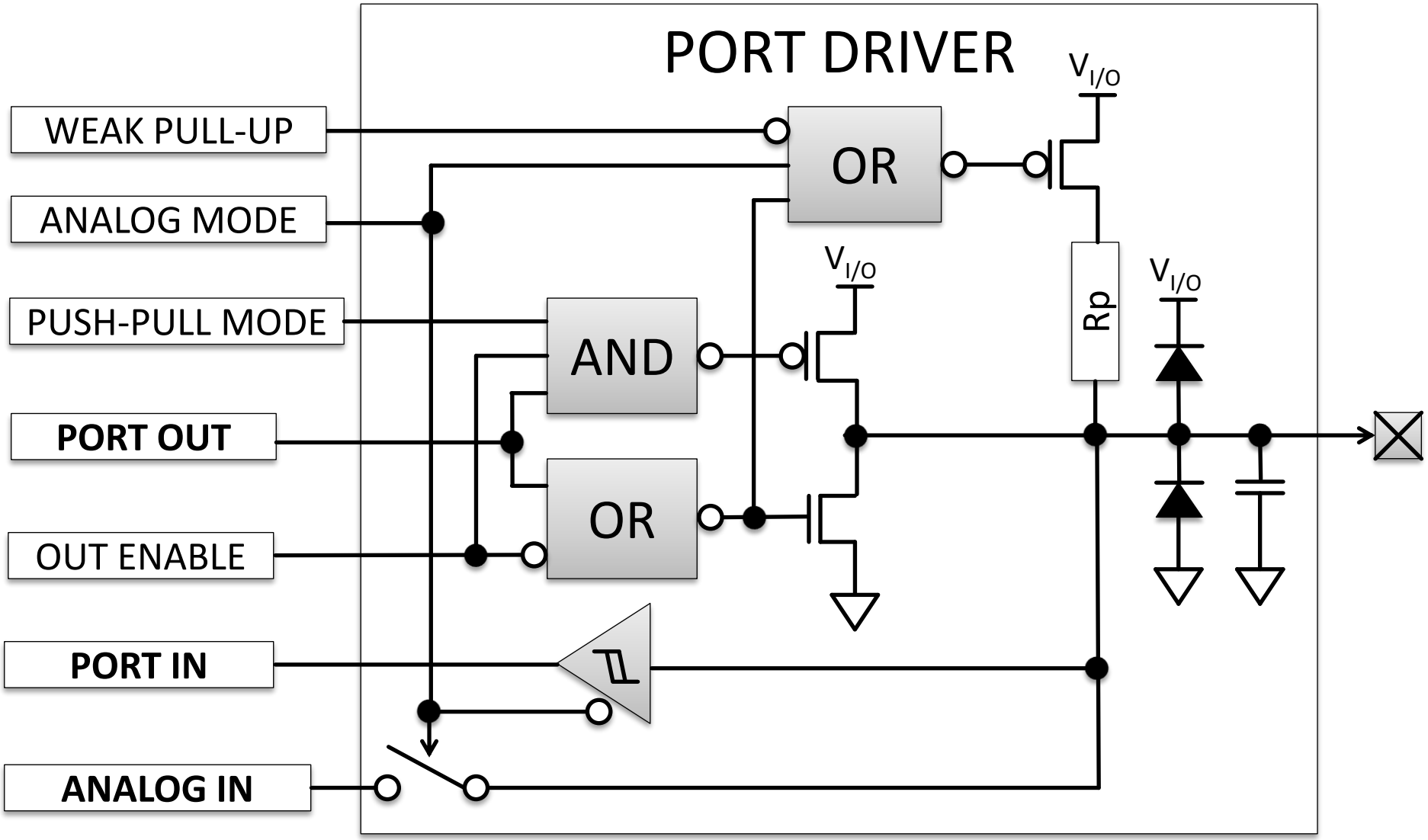
Port I/O ▸ Digital input módok



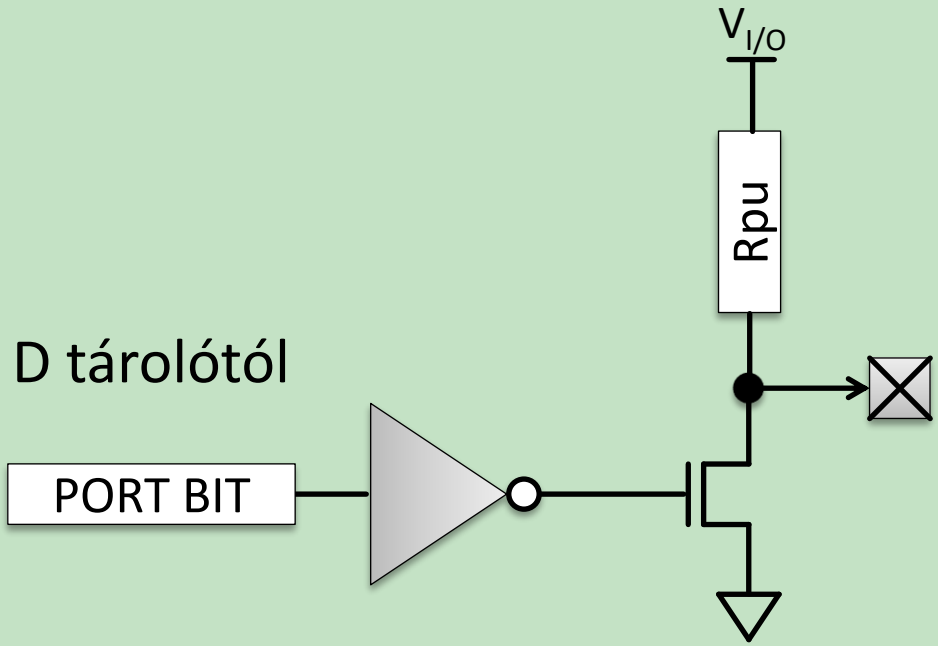
Port I/O ▸ Analog input/output



Port I/O ▸ C8051Fxxx felépítés

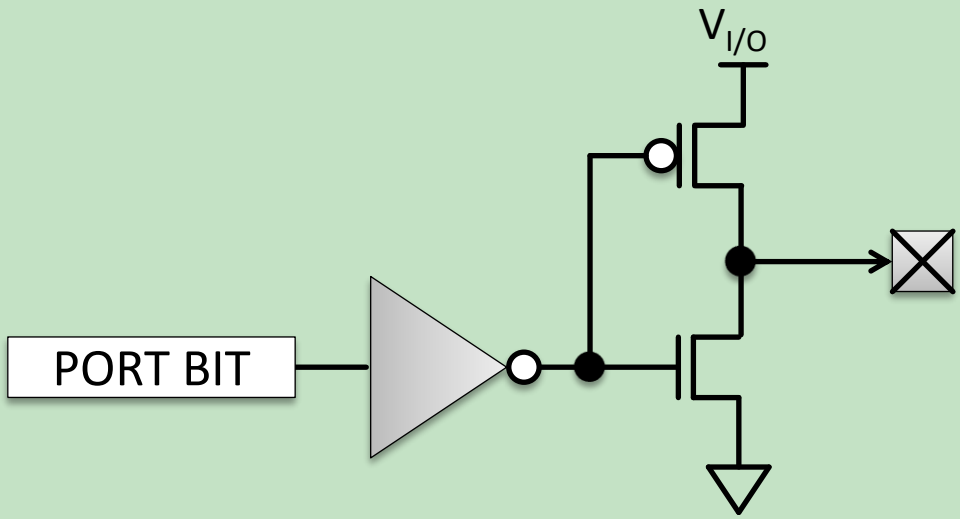


Port I/O ▸ kimeneti módok



- Alapmód: **open-drain**
  - 0: erős kötés GND-re
  - 1: belső 20k-100k $V_{I/O}$-ra
  - 1: egyben input mód is

Port I/O ▸ kimeneti módok



- **Push-pull** mód
  - 0: erős kötés GND-re
  - 1: erős kötés $V_{I/O}$-ra

Port I/O ▸ kimeneti módok

OPEN-DRAIN MÓD

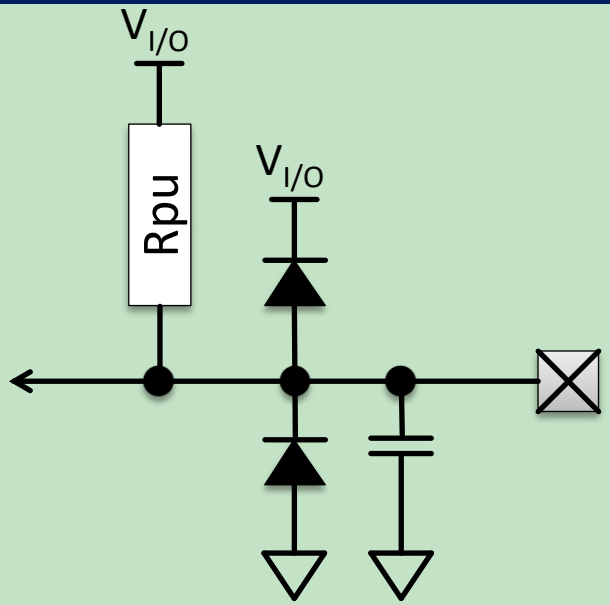
- Logikai 1 kimenet esetén
  - bementként is funkcionálhat
  - kevés perifériát lehet jól meghajtani
  - kicsi áramot tud adni a kimenet
  - Ha a kimeneten kapacitás van, lassú lehet a felfutás (≈RC)
  - Ha a kimeneten ellenállás van, könnyen elhúzhatja
- Logikai 0 kimenet esetén
  - erős meghajtásnak számít
  - nagy áramot tud húzni

PUSH-PULL MÓD

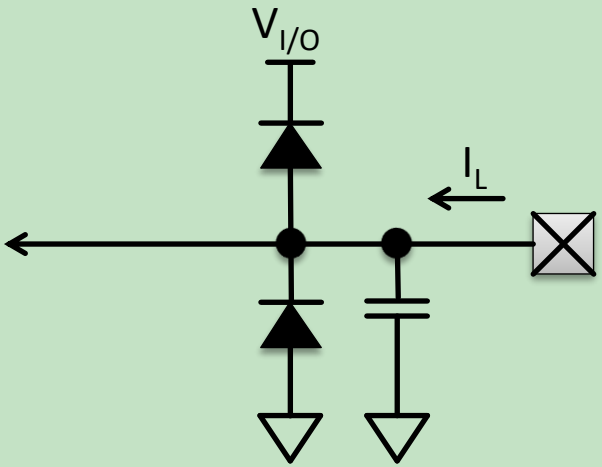
- Erős meghajtás mindkét állapot esetére
- Mindig érdemes használni, kivéve amikor open-drain szükséges (pl. wired-OR)
- Kimondottan fontos, ha
  - gyors jeleket kell előállítani
  - nagyobb áramot kell adni

Port I/O ▸ bemeneti módok

A port bitbe 1-et kell írni! Ne legyen push-pull!



- **Digitális mód**
  - Schmitt-trigger input
  - 20k-100k gyenge felhúzóellenállás
  - Diódák: ESD védelem



- **Analóg mód**
  - Kikapcsolja a felhúzóellenállást
  - Nagyimpedanciás bemenet
  - Diódák: ESD védelem

Port I/O ▸ Adatlap ▸ Absolute maximum ratings

parameter	conditions	min	typ	max	units
Voltage on any Port 0 Pin		-0.3V	-	5.5	V
Voltage on any Port I/O Pin (except Port 0 pins)		-0.3V	-	Vio+0.3	V
Maximum output current sunk by any Port pin			-	100	mA
Maximum output current sourced by any Port pin			-	100	mA
Maximum Total current through Vdd, Vio,... and GND				500	mA

Port I/O ▸ Adatlap ▸ Electrical characteristics

parameter	Conditions	min	typ	max	units
Output High Voltage (Port I/O push-pull)	I_{OH} = –3 mA I_{OH} = –70 µA	$V_{I/O}$ – 0.5 $V_{I/O}$ – 50mV	-	-	V
Output Low Voltage	**$V_{I/O}$ = 2.0 V:** I_{OL} = 70 µA I_{OL} = 8.5 mA **$V_{I/O}$ = 4.0 V:** I_{OL} = 70 µA I_{OL} = 8.5 mA	-	-	 50 800 40 400	mV
Input High Voltage		$V_{I/O}$ x 0.7	-	-	
Input Low Voltage		-	-	$V_{I/O}$ x 0.3	
Input Leakage Current	Weak Pullup Off	-	<0.1	±1	µA
Weak Pullup Impedance		-	120	-	kΩ

Port I/O ▸ Az adatlapi feltételek betartása

- Mindig tartsuk be a feltételeket!
- Ha megsértjük az **ABSOLUTE MAXIMUM RATINGS** feltételeket
  - az áramkör károsodhat
  - az áramkört azonnal selejtezni kell
  - nem szabad egy próbateszttel „megmenteni”
- Ha megsértjük az **ELECTRICAL CHARACTERISTICS** feltételeket
  - az áramkör helyes működése nem garantált
  - más részfunkciók is hibásan működhetnek
  - maradéktalanul meg kell szüntetni az okokat

Port I/O ▸ Az adatlapi feltételek megsértése

BEMENŐ FESZÜLTSÉG

- >$V_{I/O}$: áram a táp felé
- <GND: áram a 0V-tól
- terheli a jelforrást
- ha nem korlátos az áram, kárt okoz
- soros ellenállás védhet
  - érték választása
- külső áramkörös védelem

A PORT TÚLTERHELÉSE

- Nem megfelelő logikai szint jöhet létre
- Hibás működés
- A portok összárama a $V_{I/O}$ és GND áramához járul
- A $V_{I/O}$ és GND kivezetések árama korlátos!
- Károsodás történhet
- soros ellenállás védhet
  - érték választása

Port I/O ▸ Megfontolások

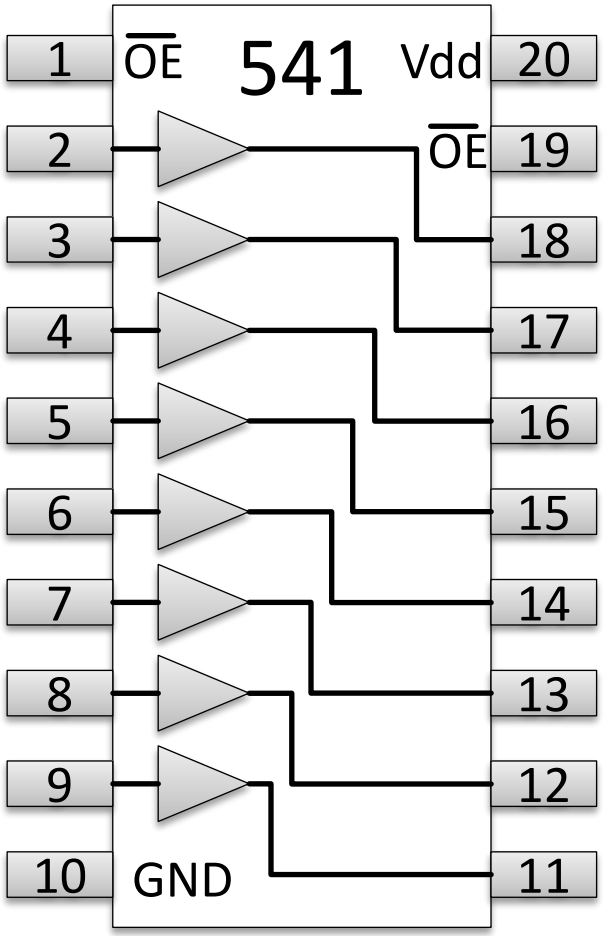
- RESET-kor a port I/O még nem aktív
  - Weak-pullup mode
  - A külső elektronika ezt figyelembe kell vegye!
    - motorvezérlés
    - külső áramkörök fals jeleket kaphatnak
    - megfelelő inicializálás kellhet a többi áramkörnek
- A crossbar engedélyezése szükséges a portok használathoz
  - Különben a kimeneti meghajtók inaktívak
  - Bemenetként enélkül is használhatók (open drain mód)

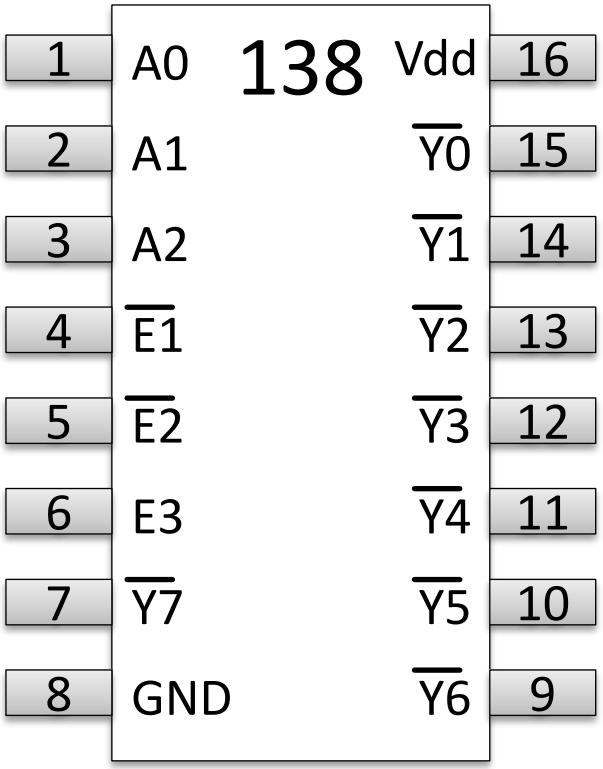
PORT I/O bővítése külső áramkörökkel

Port I/O ▸ Nagyobb kimeneti áram?

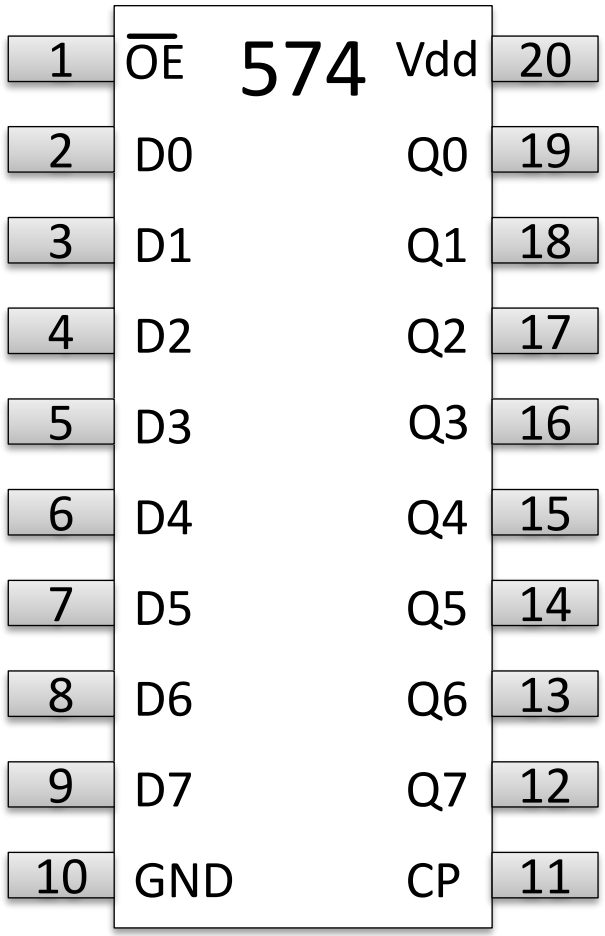
- $I_{I/O}$ korlát → külső meghajtók bővíthetik
  - pl. 74AHC541, 74AHC125, 74AHC138
- Melyik logikai család megfelelő?
  - LS, HC, HCT, AHC, AHCT, LVC?
  - A tápfeszültségtől függ, amik a logikai szinteket is meg határozzák.
- 5V vagy 3,3V is lehet a $V_{I/O}$ tépfeszültség
- Ha még nagyobb áram szükséges
  - Tranzisztor (bipoláris, MOSFET)
  - tranzisztortömb (pl. ULN2803)

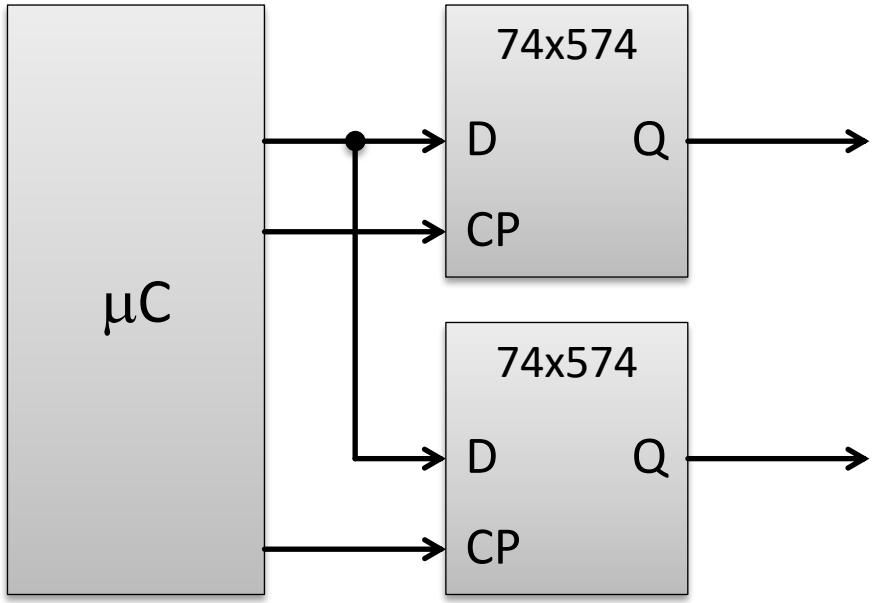
Port I/O ▸ Külső meghajtók ▸ Buffer, demultiplexer



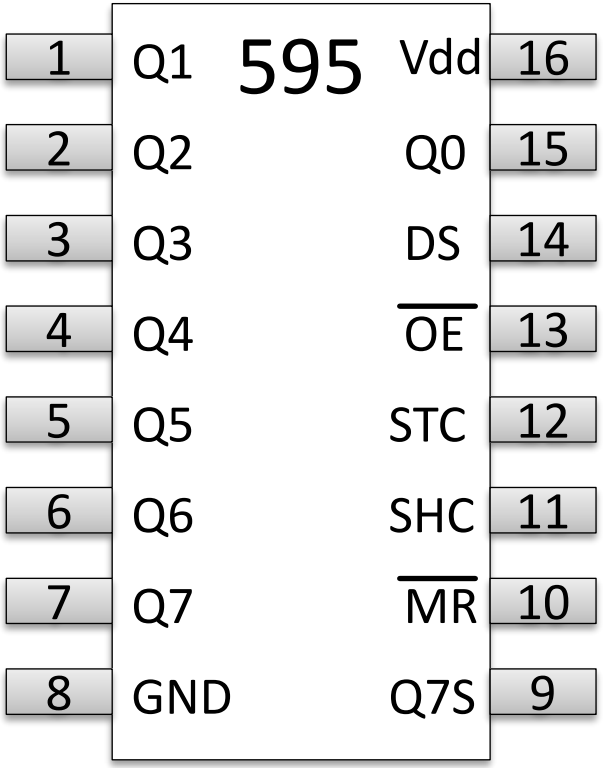


Port I/O ▸ Külső meghajtók ▸ Latch, tároló



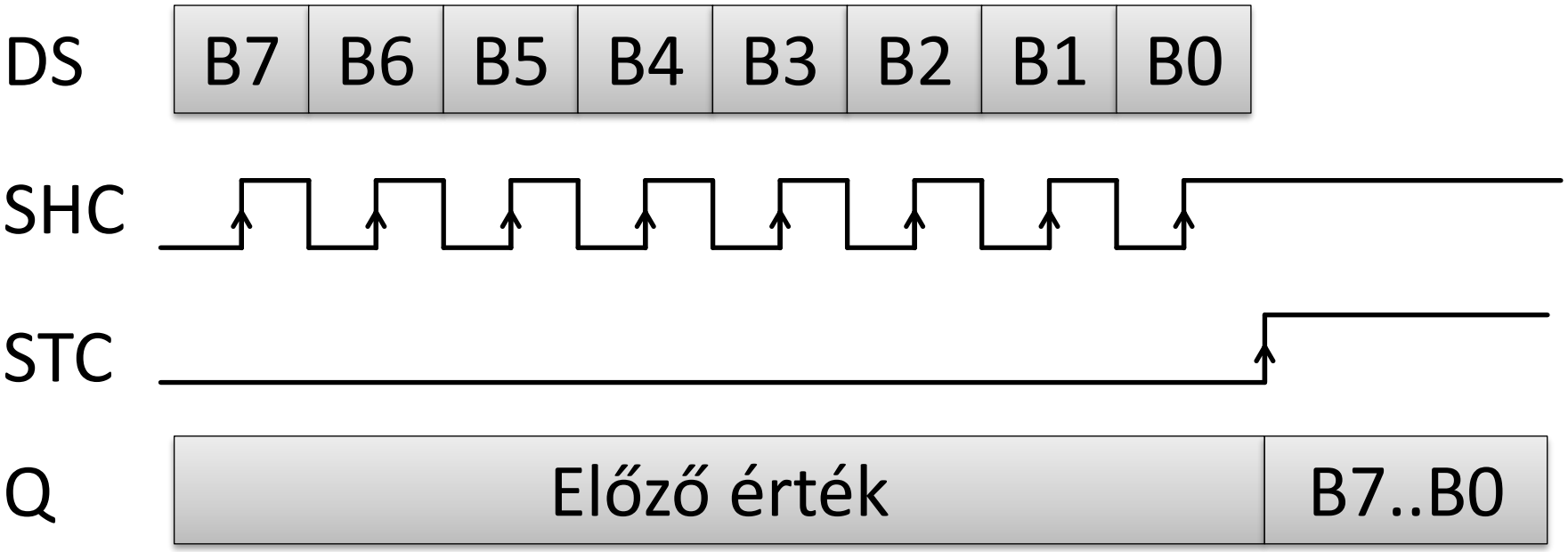


Soros-párhuzamos konverter

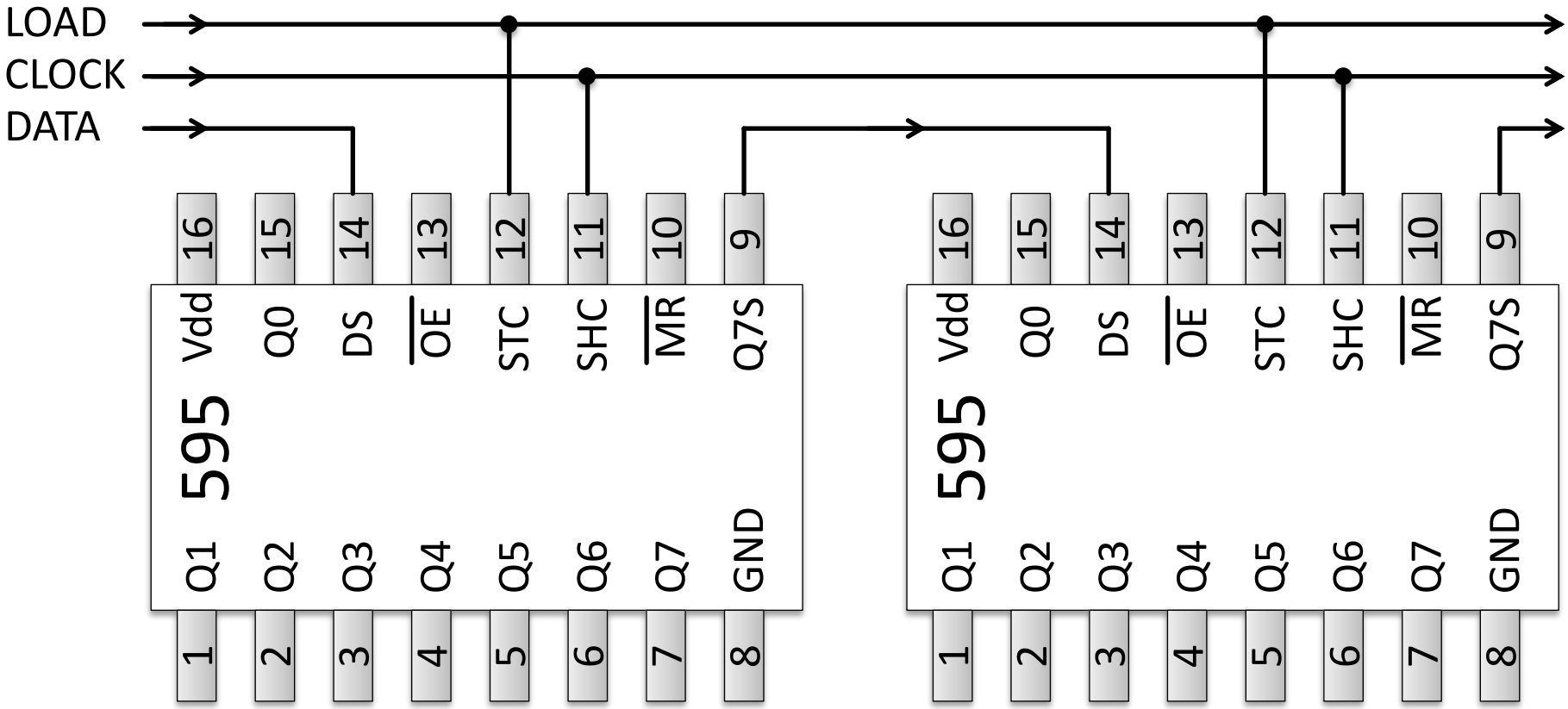


- DS: soros adat bemenet
- SHC: soros órajel
- STC: kimenetre tárolás
- Q0..Q7: kimenetek
- Q7S: soros kimenet
- MR: reset (törlés)
- OE: kimenet engedélyezés

74AHCT595

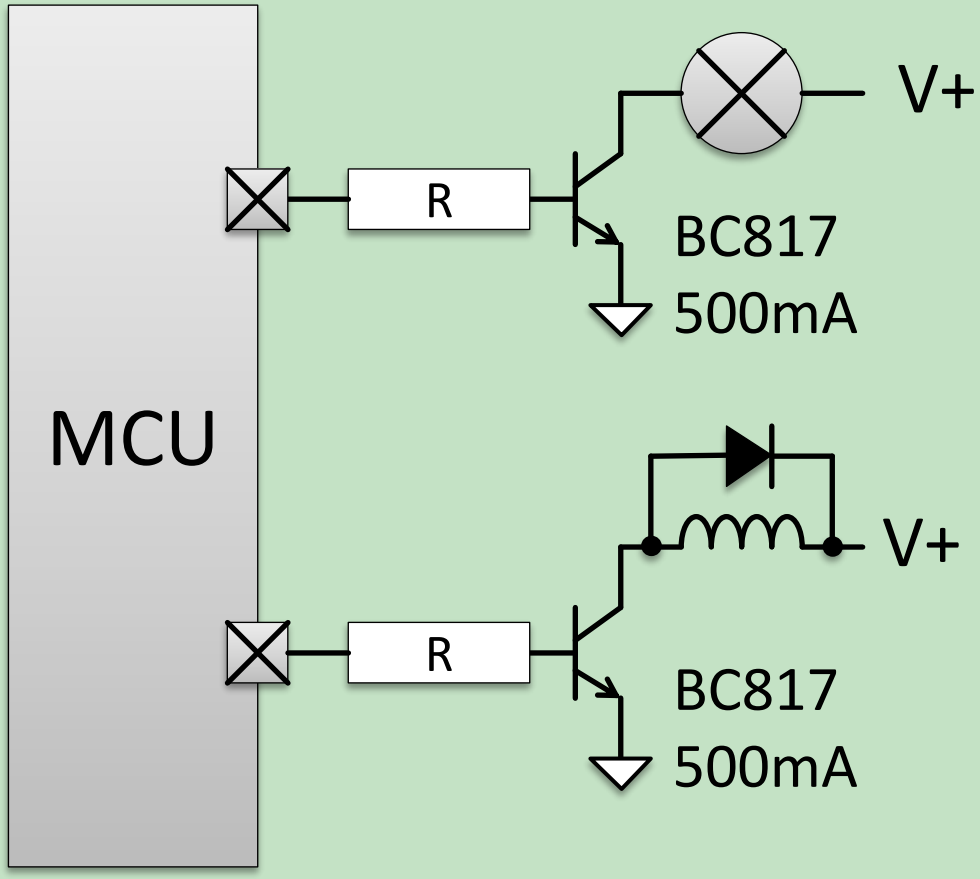


Port I/O ▸ Külső meghajtók ▸ Shift register



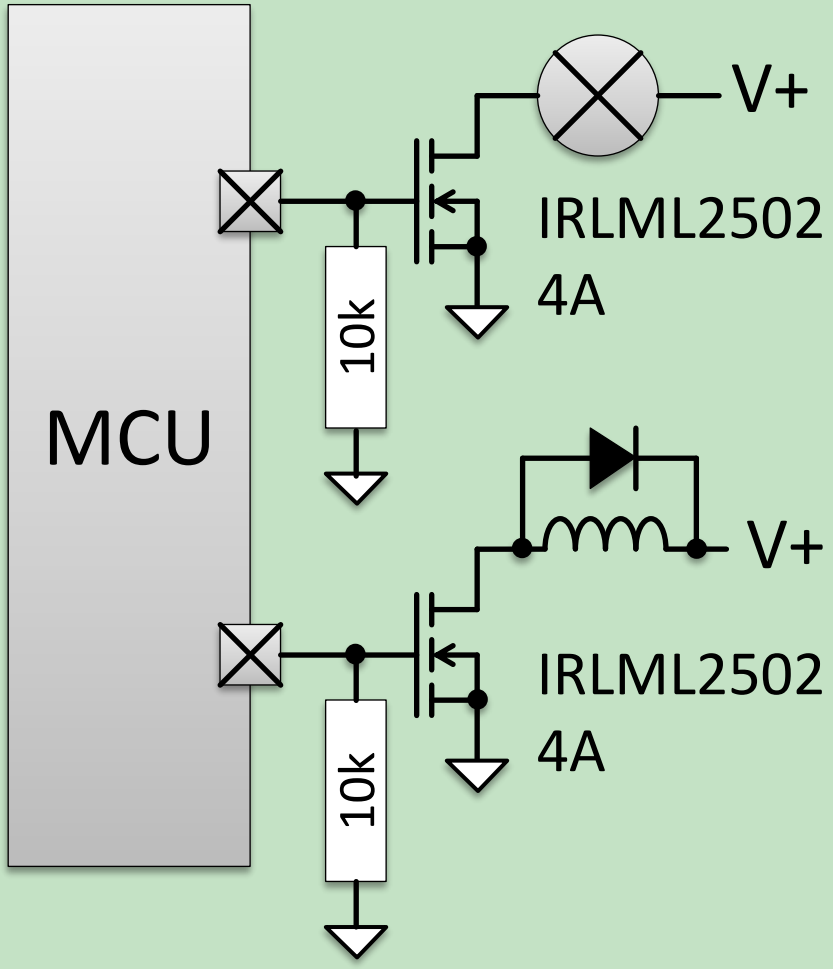
Port I/O ▸ Külső meghajtók ▸ Tranzisztorok

- A tranzisztor bázisába áram folyik
- Push-pull mód szükséges
- R szükséges

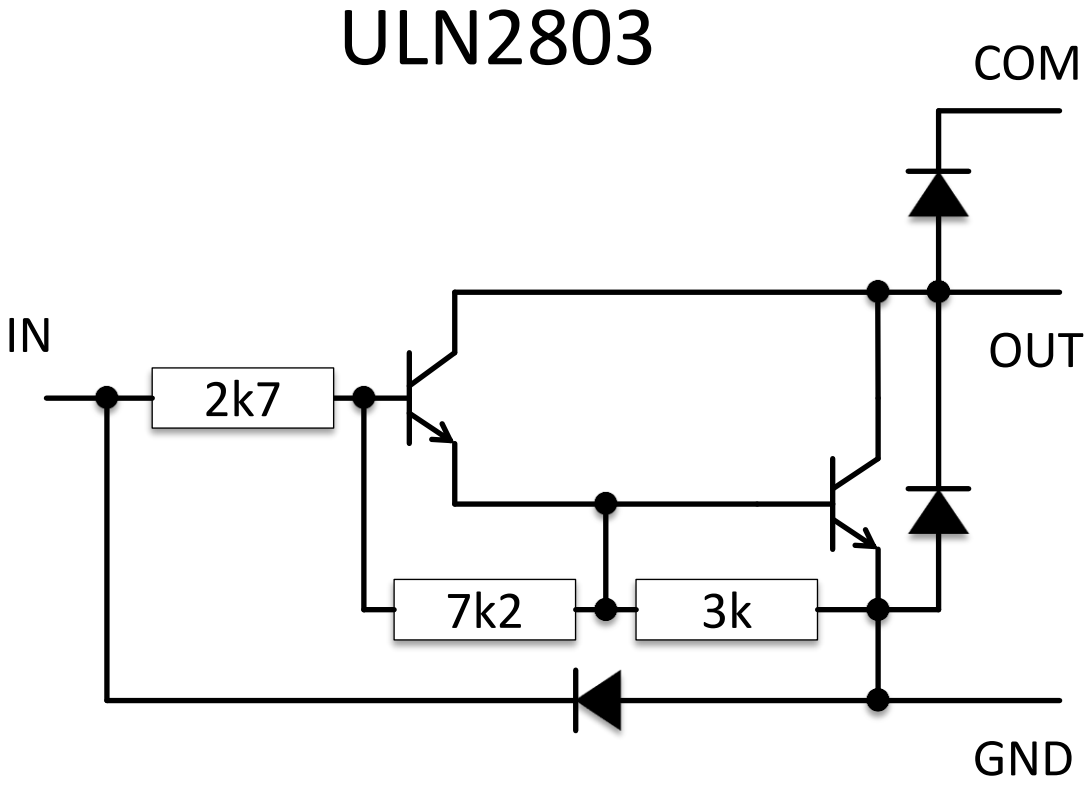


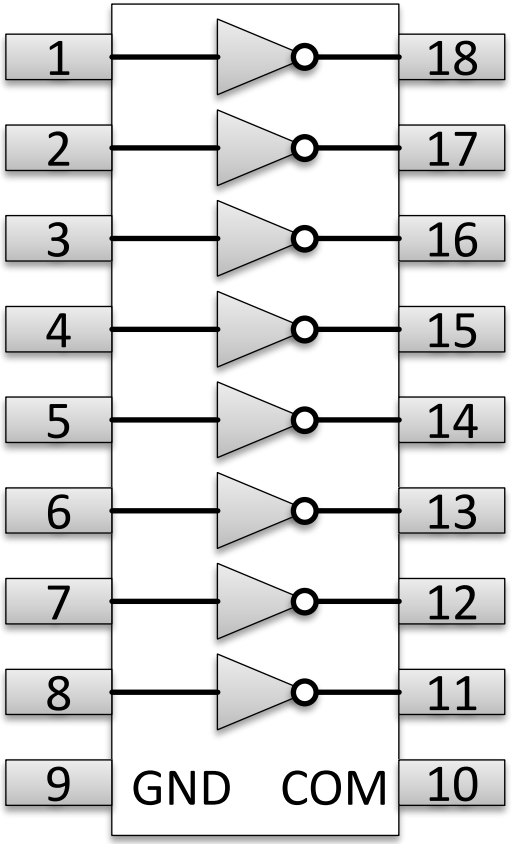
Port I/O ▸ Külső meghajtók ▸ Tranzisztorok

- Push-pull mód szükséges
  - A belső felhúzó nem elég
- A lehúzó átmeneti állapotra biztosít (pl. boot folyamatkor, open drain módban)
- Fontos a tranzisztor kapcsolási küszöbe a biztos nyitáshoz



Port I/O ▸ Külső meghajtók ▸ Tranzisztortömb





PORT I/O
Felhasználói felület – bemenet

Port I/O ▸ felhasználói felület illesztése

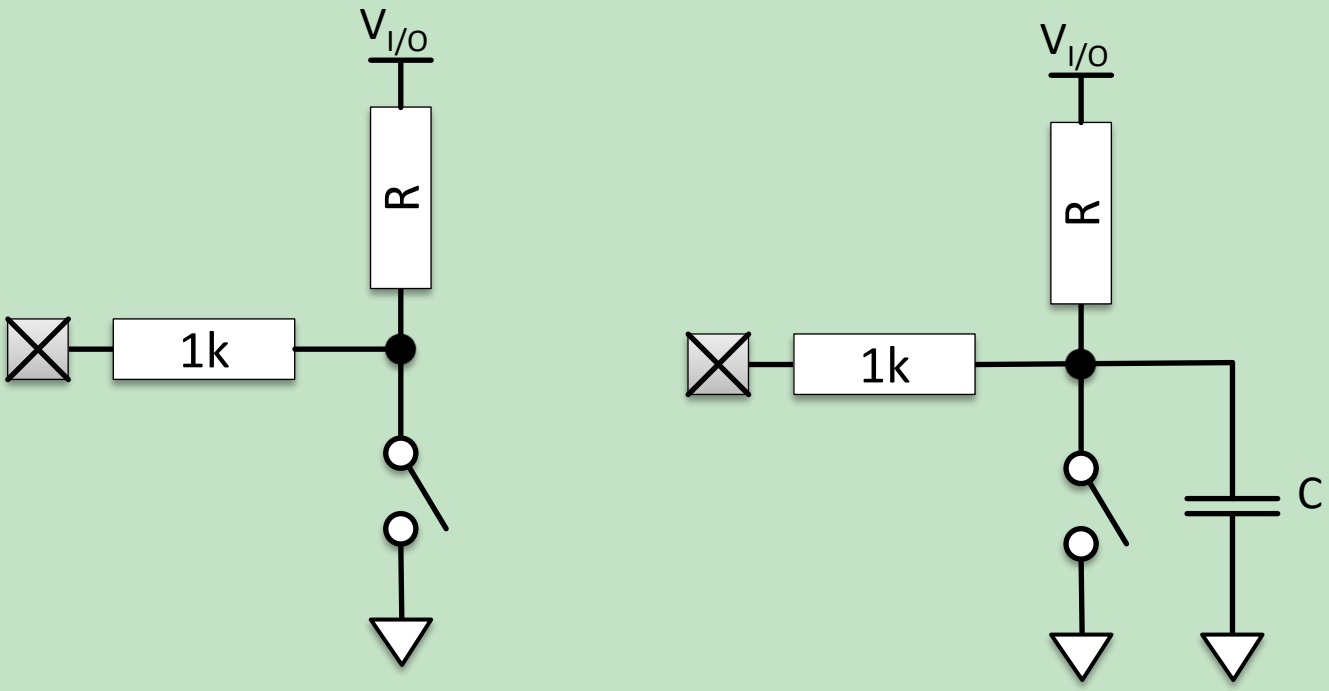
- A mikrovezérlőt tartalmazó eszközt gyakran a felhasználó beállíthatja, a működést ellenőrizheti
- Szükség van adatok
  - bevitelére
  - megjelenítésére
- Általában digitális adatokról van szó, ezek gyakran lebonthatók kétállapotú elemekre

Port I/O ▸ nyomógomb, kapcsoló

- A portokon csak digitális jel olvasható
- A nyomógomb és kapcsoló rövidzár vagy szakadás
- Ebből kell logikai 0 és 1 értéknek megfelelő feszültséget előállítani
- Egyszerű mód:
  - tápfeszültség (logikai 1) kötünk egy ellenállást és vele sorba a nyomógombot, amit a GND pontra (logikai 0) kötünk
  - benyomott állapot: 0, kiengedett: 1

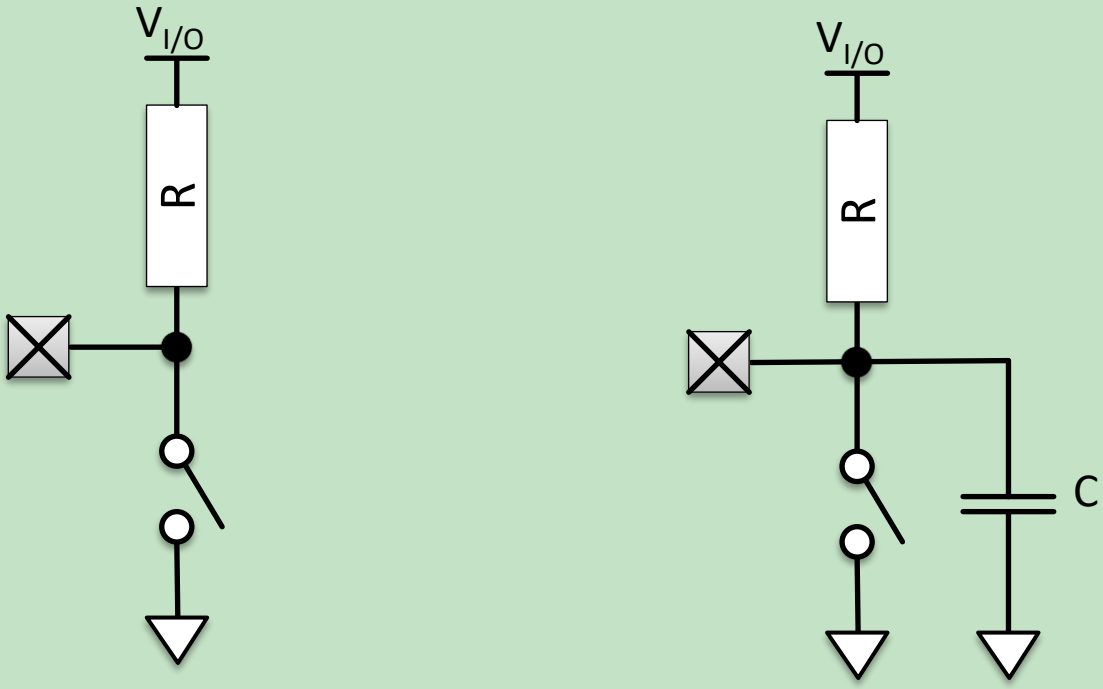
Port I/O ▸ nyomógomb, kapcsoló

- Mekkora legyen a felhúzó ellenállás (R) ?
- Szokásos értékek: **4k7**, 10k, 3k3, 2k2, 1k
- A kondenzátor csökkenti a zavarérzékenységet
- Soros 1k: a port hibás konfigurálása elleni védelem



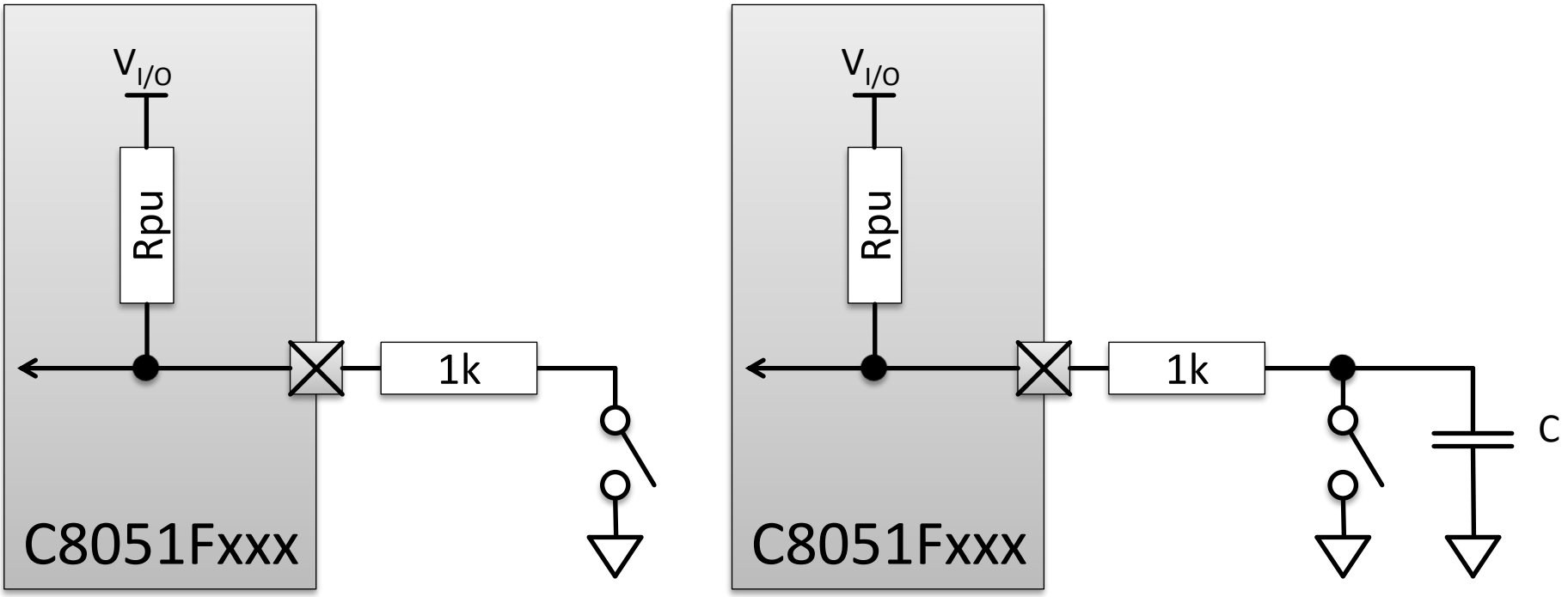
Port I/O ▸ nyomógomb, kapcsoló

- A soros védőellenállás elhagyható
  - Ha nem általános célú, sokszor újrakonfigurált a hardver
  - Gyártásra alkalmas eszközökben



Port I/O ▸ nyomógomb, kapcsoló

- Mekkora legyen az ellenállás?
- A portokon levő felhúzó ellenállás is használható
  - Zavarérzékenyebb
  - Kondenzátor segít



Port I/O ▸ nyomógomb, kapcsoló ▸ prell?

- A kapcsolás pillanatában zavarjelek keletkeznek
- Sok, gyors parazita kapcsolgatás (pergés, prell)
- Elv: rövid idejű ingadozások kiküszöbölése
  - Hardveres szűréssel (pl. kondenzátorral)
  - Szoftveres
    - a prell rövid idejű, a lenyomás viszont nem
    - a túl gyakori változásokat kell figyelmen kívül hagyni

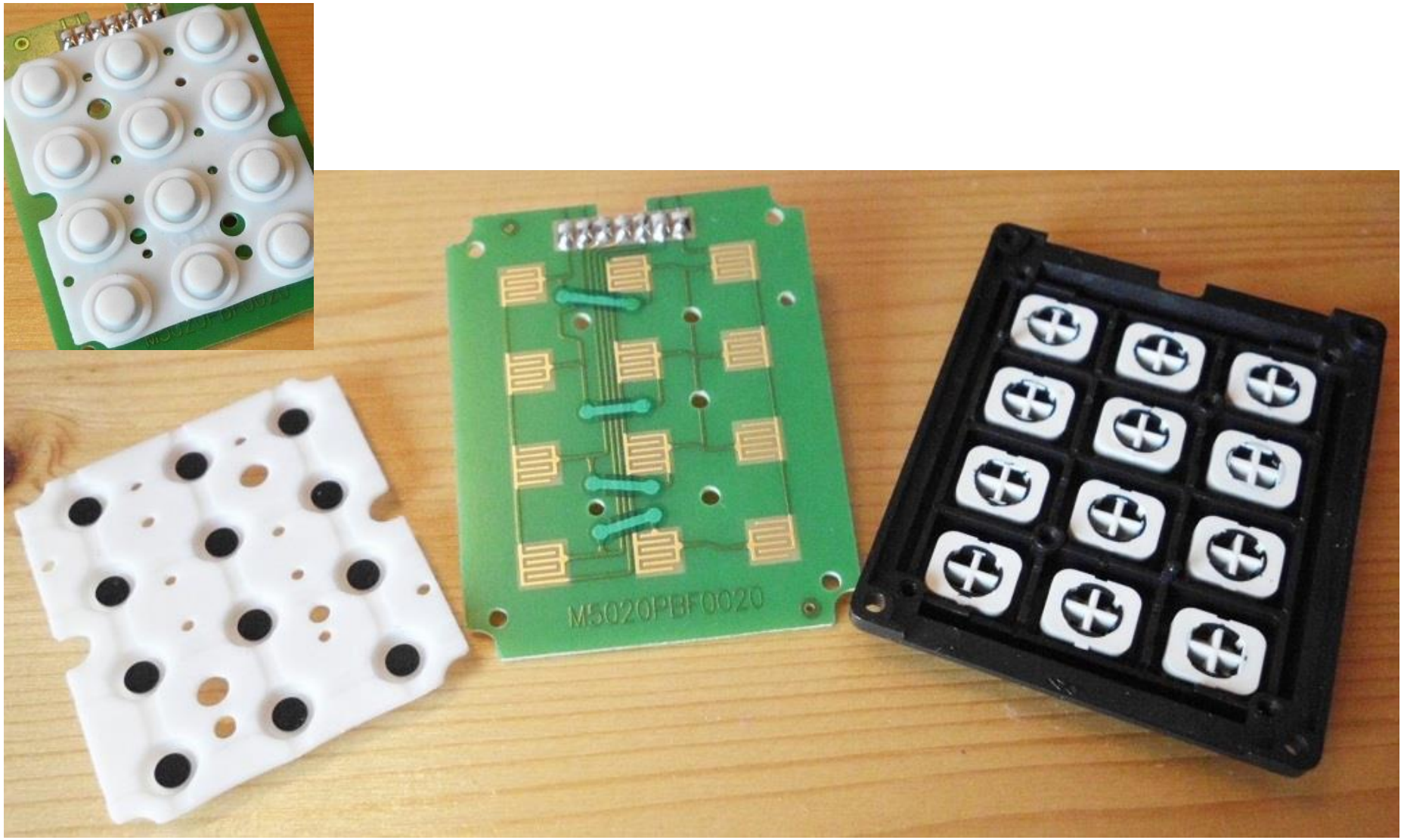
Port I/O ▸ gombnyomás detektálása

- A gomb hatására feladat végrehajtás
  - vagy a lenyomás pillanatában
  - vagy a felengedés pillanatában
  - vagy amíg a gomb nyomva van
  - ezek kiegészíthető holtidőkkel, a gombnyomások számától függő paraméterekkel, stb.
- Megfontolásra:
  - Milyen gondot okoz ezeknél a pergés?
  - Hogyan kerülhető el az egyes esetekben?

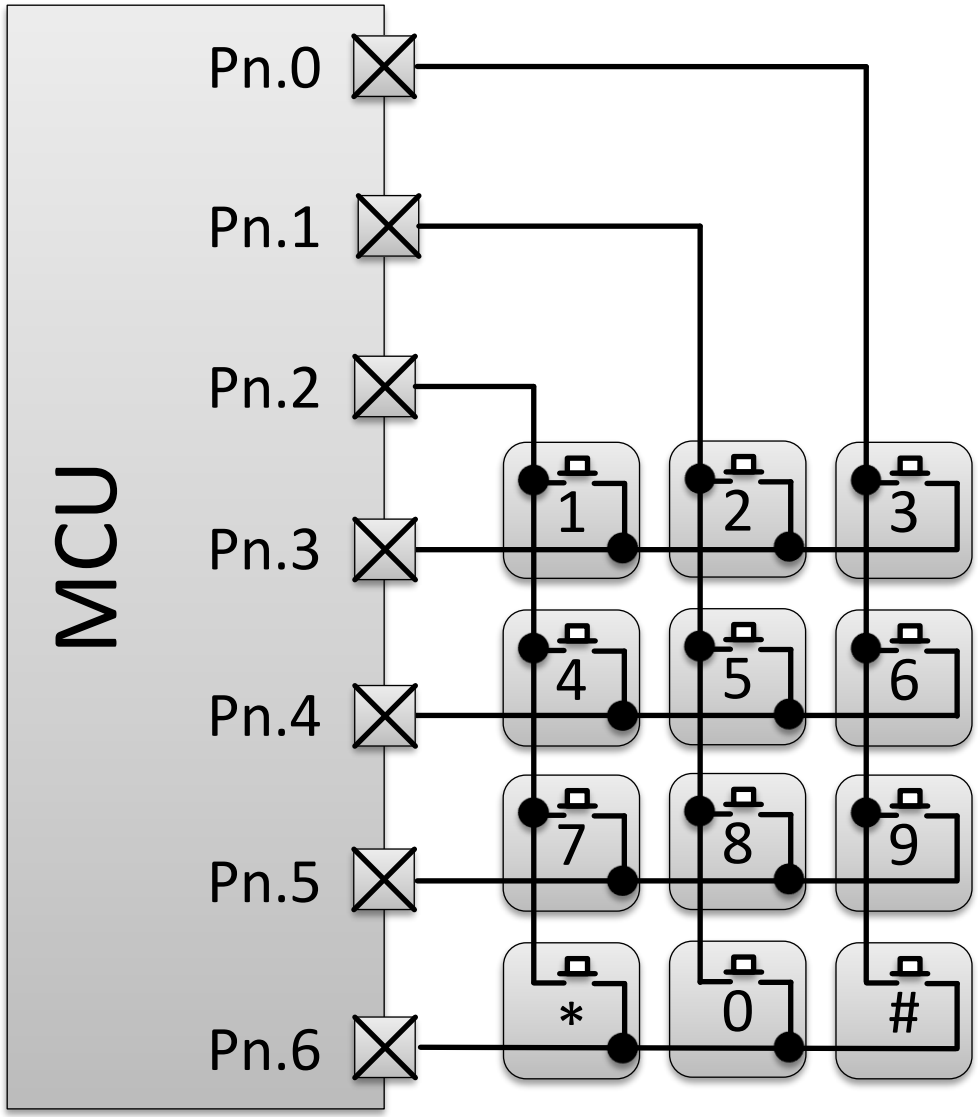
Port I/O ▸ billenytűmátrix

- Sok nyomógomb beolvasása
  - telefonbillenytűzet
  - point-of-sale (POS) terminál
  - automated teller machine (ATM)
- A szükséges jelek redukálására mátrix
- A beolvasáshoz minden lehetőséget meg kell nézni
- Megállhatunk, ha megtaláltunk egyet
- Több is lenyomva?

Port I/O ▸ billenytűmátrix ▸ felépítés



Port I/O ▸ billenytűmátrix ▸ kapcsolás



- Pn.0-Pn.2 kimenetek
- Pn.3-Pn.6 bemenetek
- **1. oszlop tesztje**:
  - Pn.2=0, Pn.1=1, Pn.0=1
  - Pn.3-Pn.6 beolvasása
- **2. oszlop tesztje**:
  - Pn.2=1, Pn.1=0, Pn.0=1
  - Pn.3-Pn.6 beolvasása
- **3. oszlop tesztje**:
  - Pn.2=1, Pn.1=1, Pn.0=0
  - Pn.3-Pn.6 beolvasása

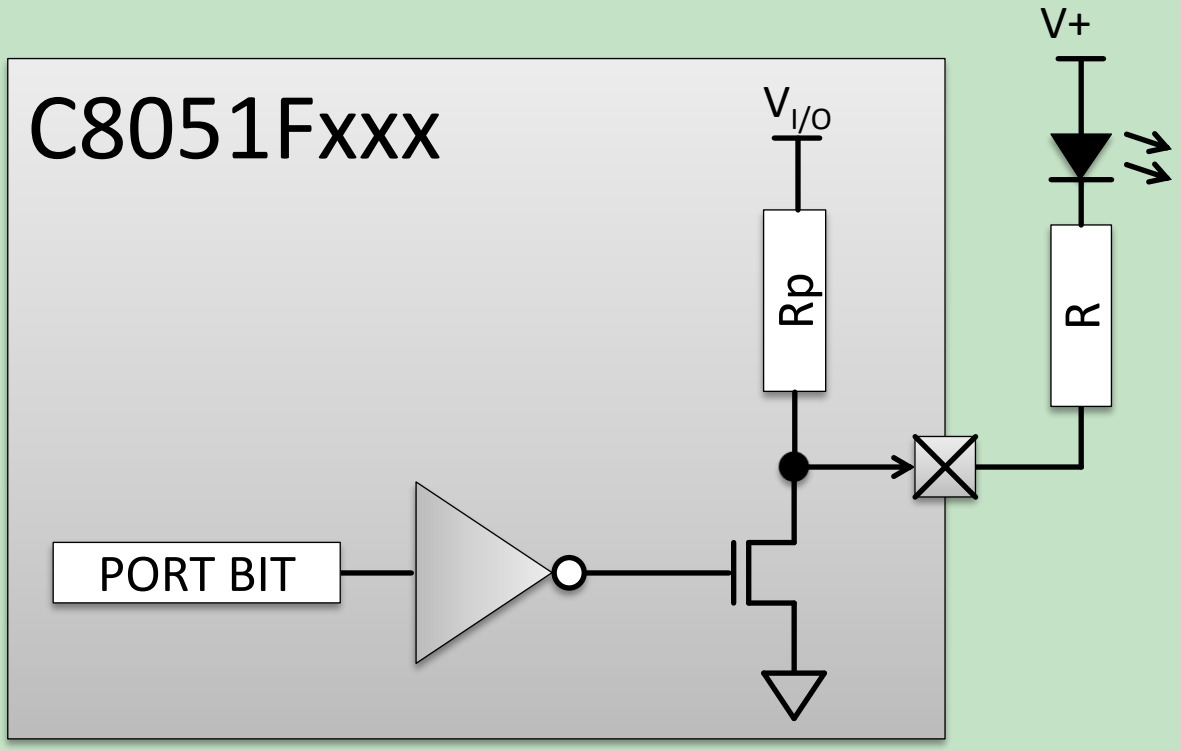
Port I/O ▸ billenytűmátrix

- Szükségesek a belső felhúzóellenállások
- Külsővel kiválthatók
  - Kisebb ellenállásérték nagyobb zavarvédettség
- Lehet a sorokra is kimenetet kötni, és az oszlopokat olvasni
- Ha több gomb van lenyomva, több bit lesz 0 olvasáskor

PORT I/O
Felhasználói felület – kimenet

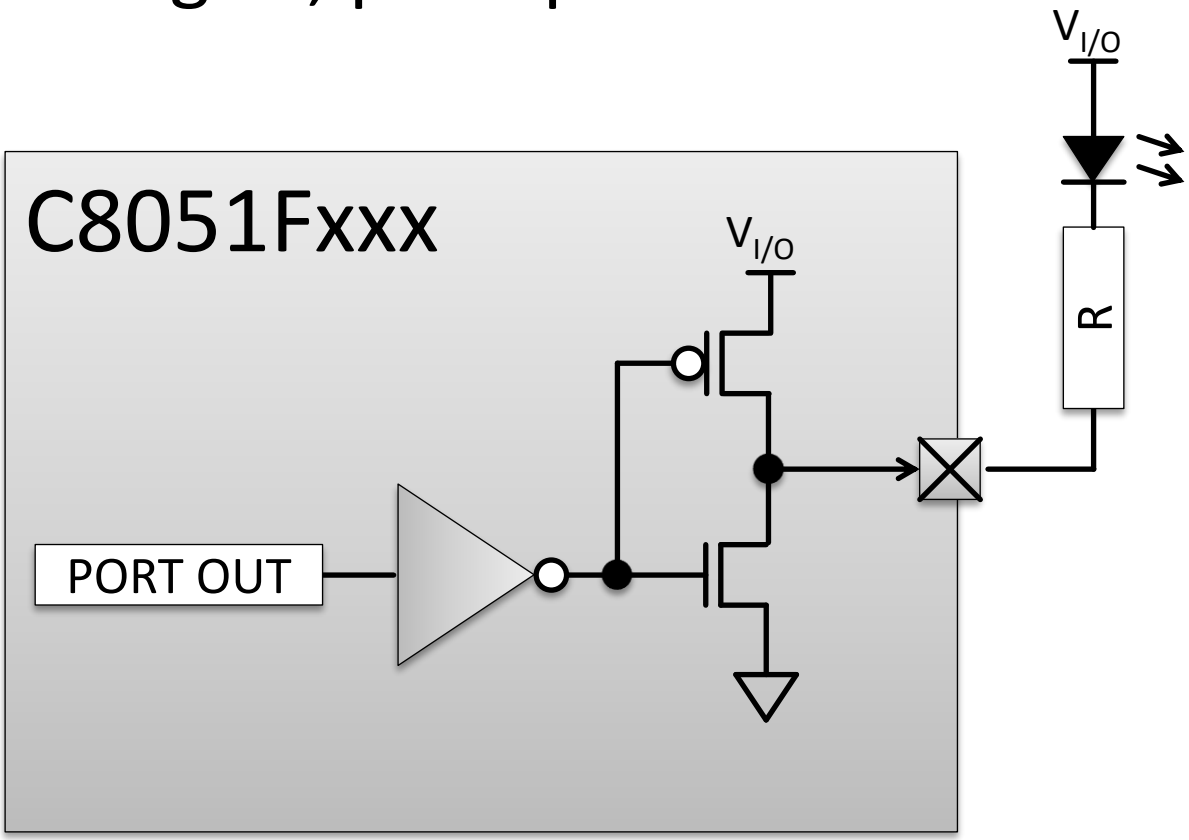
Port I/O ▸ LED

- Negatív logika, open drain mód
- A LED táp lehet akár más, mint $V_{I/O}$
- R állítja be az áramot (adatlapi korlátja van)



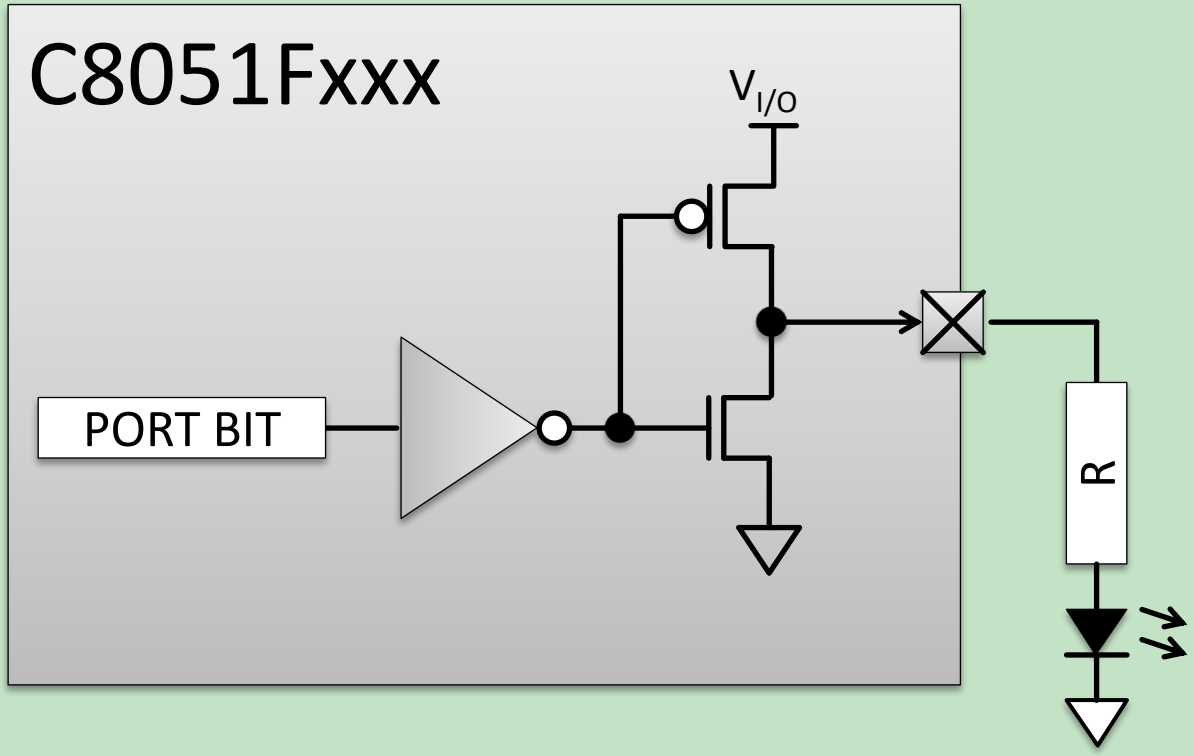
Port I/O ▸ LED

- Negatív logika, push-pull mód

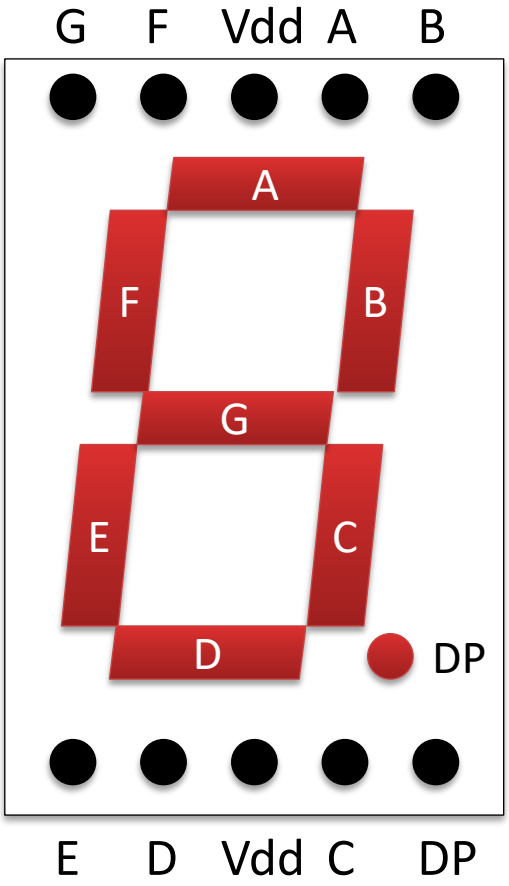


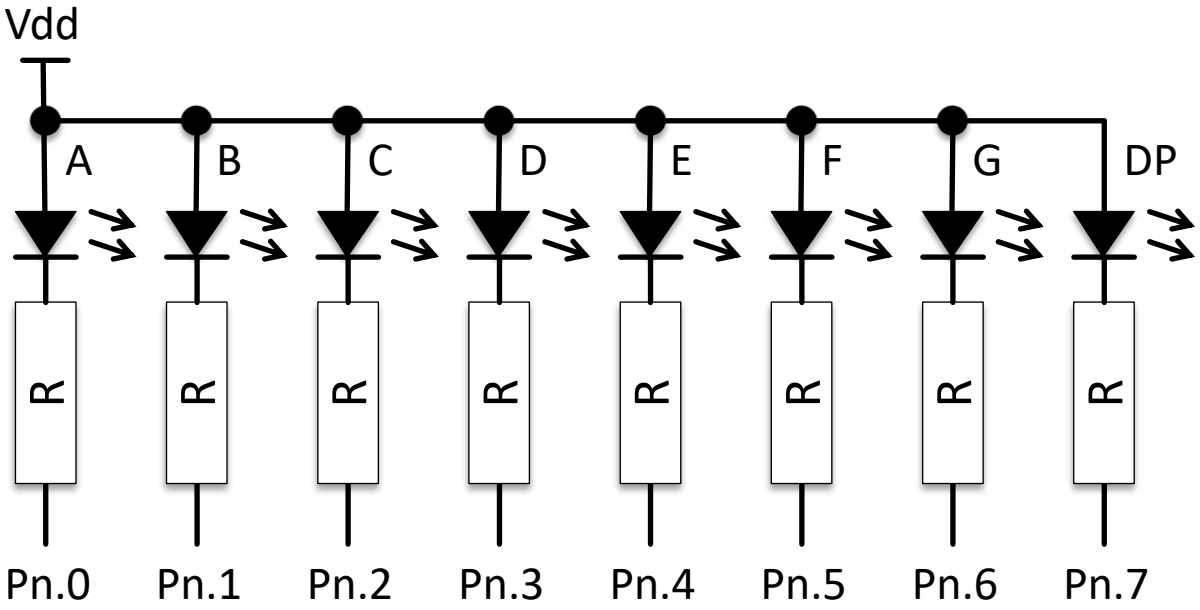
Port I/O ▸ LED

- Pozitív logika, **csak push-pull mód alkalmas**



Port I/O ▸ 7-szegmenses kijelző – közös anód





- Open drain és push-pull is jó
- Létezik közös katódú is: push-pull

Port I/O ▸ 7-szegmenses kijelző

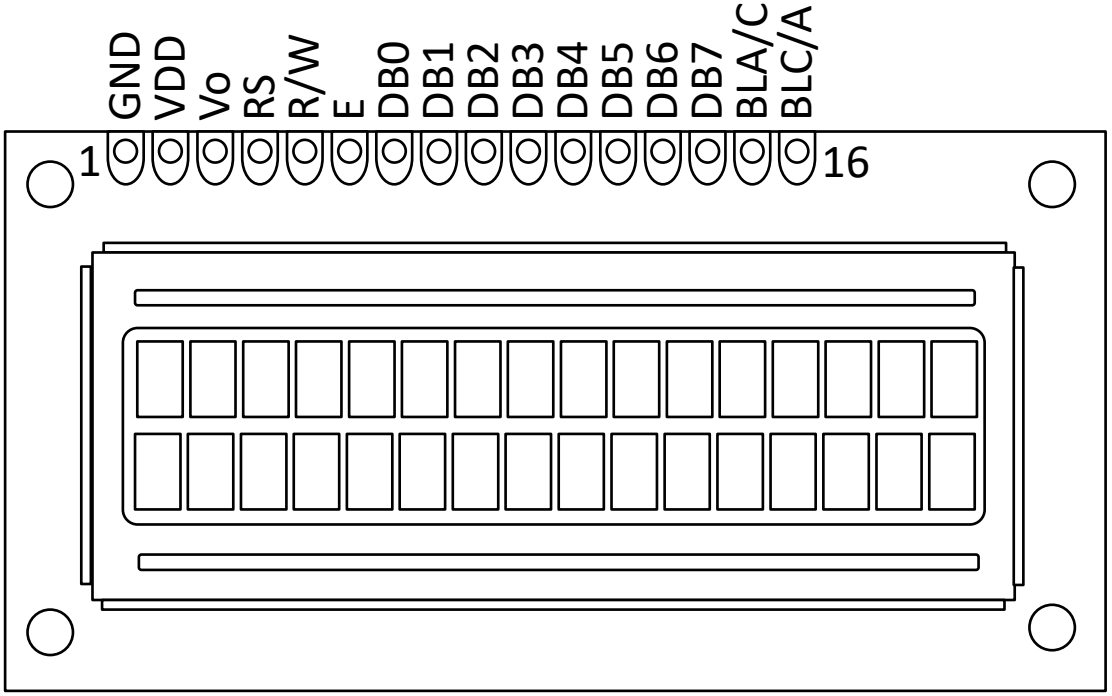
- Negatív logika: P0=~ledbits; // bytewise

BCD	G	F	E	D	C	B	A	ledbits
0	0	1	1	1	1	1	1	**3Fh**
1	0	0	0	0	1	1	0	**06h**
2	1	0	1	1	0	1	1	**5Bh**
3	1	0	0	1	1	1	1	**4Fh**
4	1	1	0	0	1	1	0	**66h**
5	1	1	0	1	1	0	1	**6Dh**
6	1	1	1	1	1	0	1	**7Dh**
7	0	0	0	0	1	1	1	**07h**
8	1	1	1	1	1	1	1	**7Fh**
9	1	1	0	1	1	1	1	**6Fh**

Port I/O ▸ 7-szegmenses kijelző

- Fényesség – nagyobb áram
  - meghajtó lehet szükséges
  - tranzisztorok vagy tranzisztortömb (pl. ULN2803A)
- Több kijelző?
  - külső latch (pl. 74HC574)
  - időosztásos kapcsolás (a fényerő csökken)

Port I/O ▸ Alfanumerikus LCD



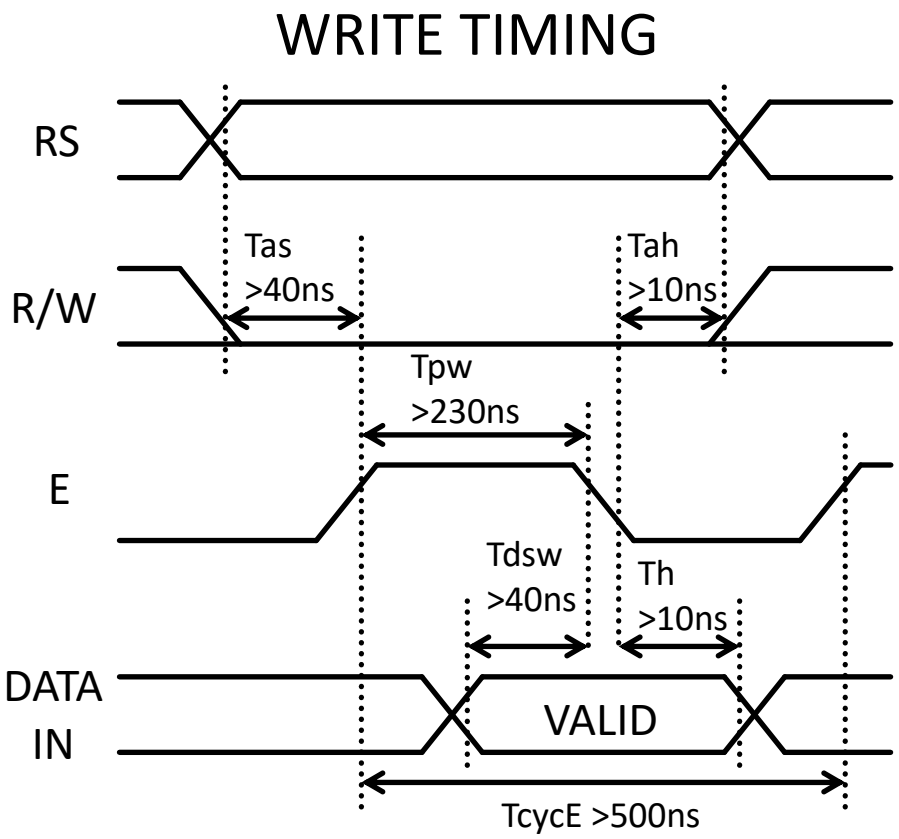
- Legelterjedtebb kiépítések:
  - 2x16 karakter
  - 2x20 karakter
  - 4x16 karakter
  - 4x20 karakter
- Opcionális háttérvilágítás

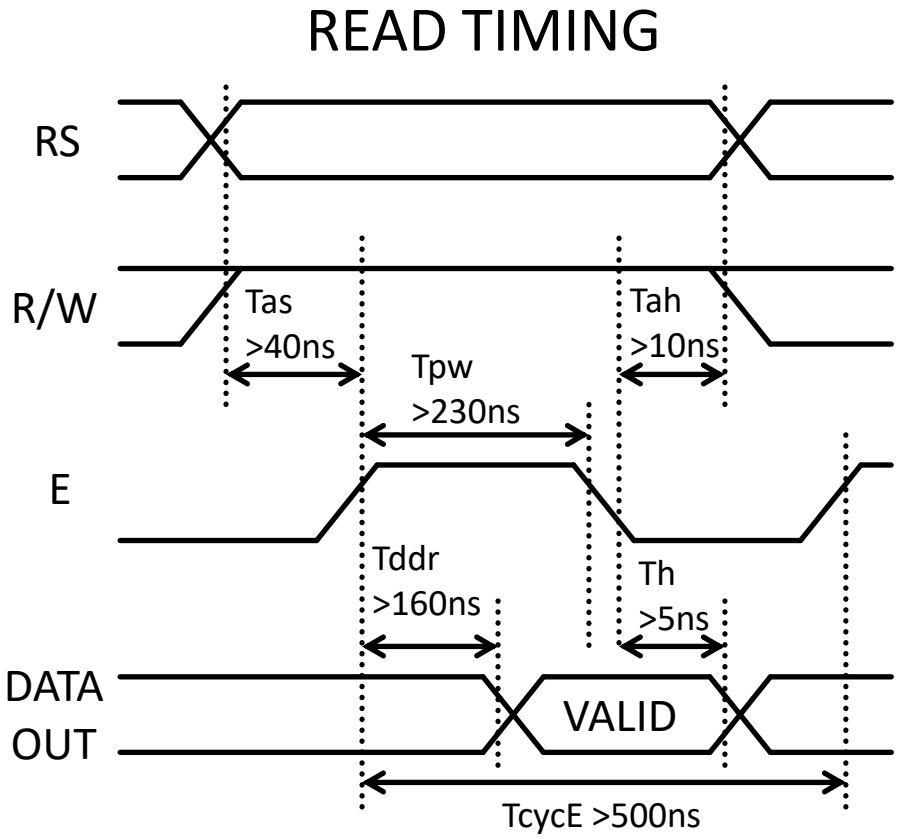
- **Táp** (bemenet)
  - GND
  - Vdd (5V vgy kisebb)
- **Vo** (bemenet)
  - kontraszt, Vdd-hez közeli
- **Vezérlőjelek** (bemenet)
  - RS: cím vagy adat
  - RW: olvasás, írás
  - E: engedélyezés
- **Adatbusz** (kétirányú)
  - DB0-DB7
  - DB4-DB7 négybites módban
- **háttérvilágító LED** (BL)
  - Anód és katód, nem standard
  - 50mA-300mA

Port I/O ▸ Alfanumerikus LCD

- Speciális párhuzamos adatbusz
  - nincs hozzá hardver által kezelt megoldás
- Ilyenkor: „**bit banging”:**
  - Szoftveresen állítjuk be a jeleket a kívánt értékűre megfelelő időzítéssel

Port I/O ▸ Alfanumerikus LCD ▸ I/O





- **E** lefutó éle ír be
- Érvényes a lefutó élnél

- **E** felfutó éle indít olvasást
- Érvényes a lefutó élnél

Port I/O ▸ Alfanumerikus LCD ▸ Regiszterek

UTASÍTÁSREGISZTER ELÉRÉSE

- **RS=0**
- Inicializálás
- Üzemmód beállítása
- Kijelző törlése
- Kurzorpozíció beállítása
- Villogó kurzor
- Automatikus kurzorléptetés
- …

ADATREGISZTER ELÉRÉSE

- **RS=1**
- karakterek írása a kijelzőre
- aktuális kurzorpozícióra
- Karakter kódjának írásával
- Egysoros kijelzők kurzorcímei
  - 0x00+i
- Kétsoros kijelzők kurzorcímei
  - 1. sor: 0x00+i
  - 2. sor: 0x40+i
- Négysoros kijelzők kurzorcímei
  - 1. sor: i
  - 2. sor: 0x40+i
  - 3. sor: sorhossz+i
  - 4. sor: 0x40+ sorhossz+i

Port I/O ▸ Alfanumerikus LCD ▸ Utasítások

Utasítás	adatbitek								megjegyzés
	B7	B6	B5	B4	B3	B2	B1	B0	
Clear display	0	0	0	0	0	0	0	1	A teljes kijelző törlése
Return home	0	0	0	0	0	0	1	-	A kurzor és beviteli mód alaphelyzetbe hozása
Entry mode set	0	0	0	0	0	1	ID	S	Beviteli mód: ID=1:kurzor jobbra S=1:teljes kijelző eltolás
Display on/off	0	0	0	0	1	D	C	B	Kijelző: D=1 bekapcsolás, C=1 kurzor be, B=1 kurzor villog
Cursor or display shift	0	0	0	1	SC	RL	-	-	SC=1:kijelző, 0:kurzor RL=1:jobbra, 0: balra mozgatás
Function set	0	0	1	DL	N	F	-	-	DL=1:8-bites, 0:4-bites mód N=1:2 sor, 0:1 sor F=1:5x10, 0:5x8 képpont egy betű

Port I/O ▸ Alfanumerikus LCD ▸ Utasítások

Utasítás	adatbitek								megjegyzés
	B7	B6	B5	B4	B3	B2	B1	B0	
Set CGRAM address	0	1	A5	A4	A3	A2	A1	A0	karaktertáblába írás címe saját karakterek definiálására
Set DDRAM address	1	A6	A5	A4	A3	A2	A1	A0	a kijelzőre írás címe, azaz a kurzor pozicionálása

- RS=0 és R/W=0 kell az utasítások küldéséhez
- Az utasítások valamennyi ideig futnak (~40us)
- Ez alatt nem szabad a kijelzőbe írni!
- Megoldás:
  - megfelelő szoftveres késleltetés az utasítás kiadása után
  - az állapotbit beolvasása (B7), R/W=1 állításával

Port I/O ▸ Alfanumerikus LCD ▸ 8-bites mód

```
#define LCD_RS   P0.5
#define LCD_RW   P0.6
#define LCD_E    P0.7
#define LCD_PORT P1

unsigned char line_address[4];

void LCD_Init(unsigned char rows, unsigned char cols)
{
  unsigned char i;

  line_address[0]=0;
  line_address[1]=0x40;
  line_address[2]=cols;
  line_address[3]=0x40+cols;
```

Port I/O ▸ Alfanumerikus LCD ▸ 8-bites mód

```
  LCD_RW=0;
  LCD_E=0;
  LCD_RS=0;
  Delay_ms(50);     // Special initialization sequence
  LCD_DATA=0x30;    // 8-bit mode
  LCD_PulseE();
  Delay_ms(5);
  LCD_PulseE();     // 8-bit mode
  Delay_ms(1);
  LCD_PulseE();     // 8-bit mode
  LCD_Write(0x38); // 8-bit mode, 2 lines
  LCD_Write(0x08); // display off
  LCD_Write(0x01); // display clear
  LCD_Write(0x06); // entry mode: increment cursor
  LCD_Write(0x0C); // display on, no cursor, no blink
}
```

Port I/O ▸ Alfanumerikus LCD ▸ 8-bites mód

```
void LCD_PulseE(void)
{
  unsigned char i;
  for(i=0;i<100;i++);
  LCD_E=1;
  for(i=0;i<100;i++);
  LCD_E=0;
}

void LCD_Write(unsigned char a)
{
  LCD_DATA=a;
  LCD_PulseE();
  Delay_ms(2);   // or check busy flag
}
```

Port I/O ▸ Alfanumerikus LCD ▸ 8-bites mód

```
void LCD_Clear(void)
{
  LCD_RS=0;
  LCD_Write(1);
  LCD_RS=1;
}

void LCD_MoveTo(unsigned char line, unsigned char pos)
{
  LCD_RS=0;
  LCD_Write(0x80 | (line_address[line]+pos));
  LCD_RS=1;
}
```

Port I/O ▸ Alfanumerikus LCD ▸ 4-bites mód

- Két részletben lehet írni a regisztereket
  - A felső négy bitet először
  - Az alsó négy bitet ezután
- A D7..D4 adatbiteket kell használni

Port I/O ▸ Alfanumerikus LCD ▸ C stdout

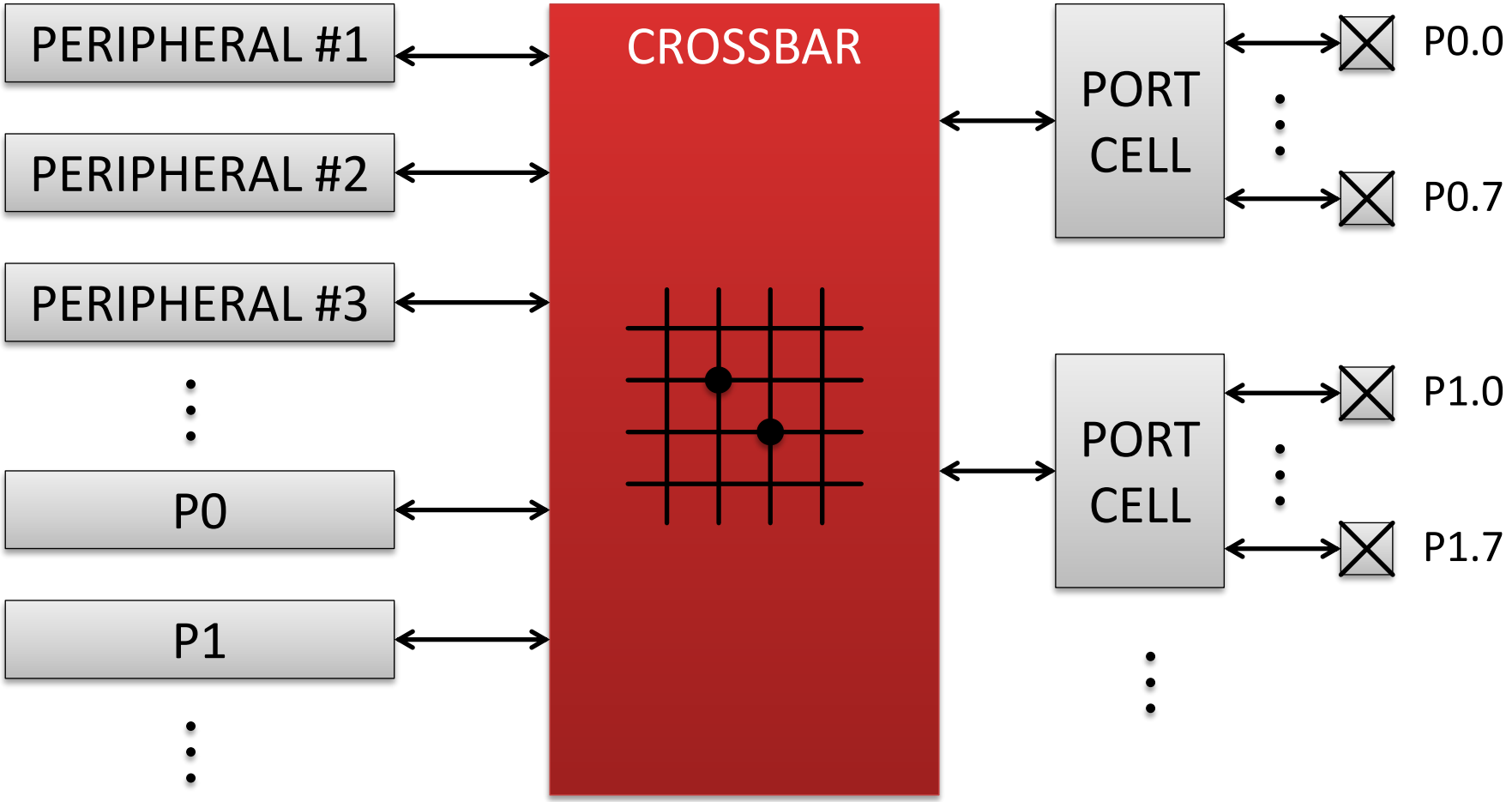
- stdio átirányítás? printf?

```
void putchar(char c)
{
    LCD_Write(c);
}
…
LCD_MoveTo(0,10);   // first line, 10th position
printf("x=%d",x);   // write
…
```

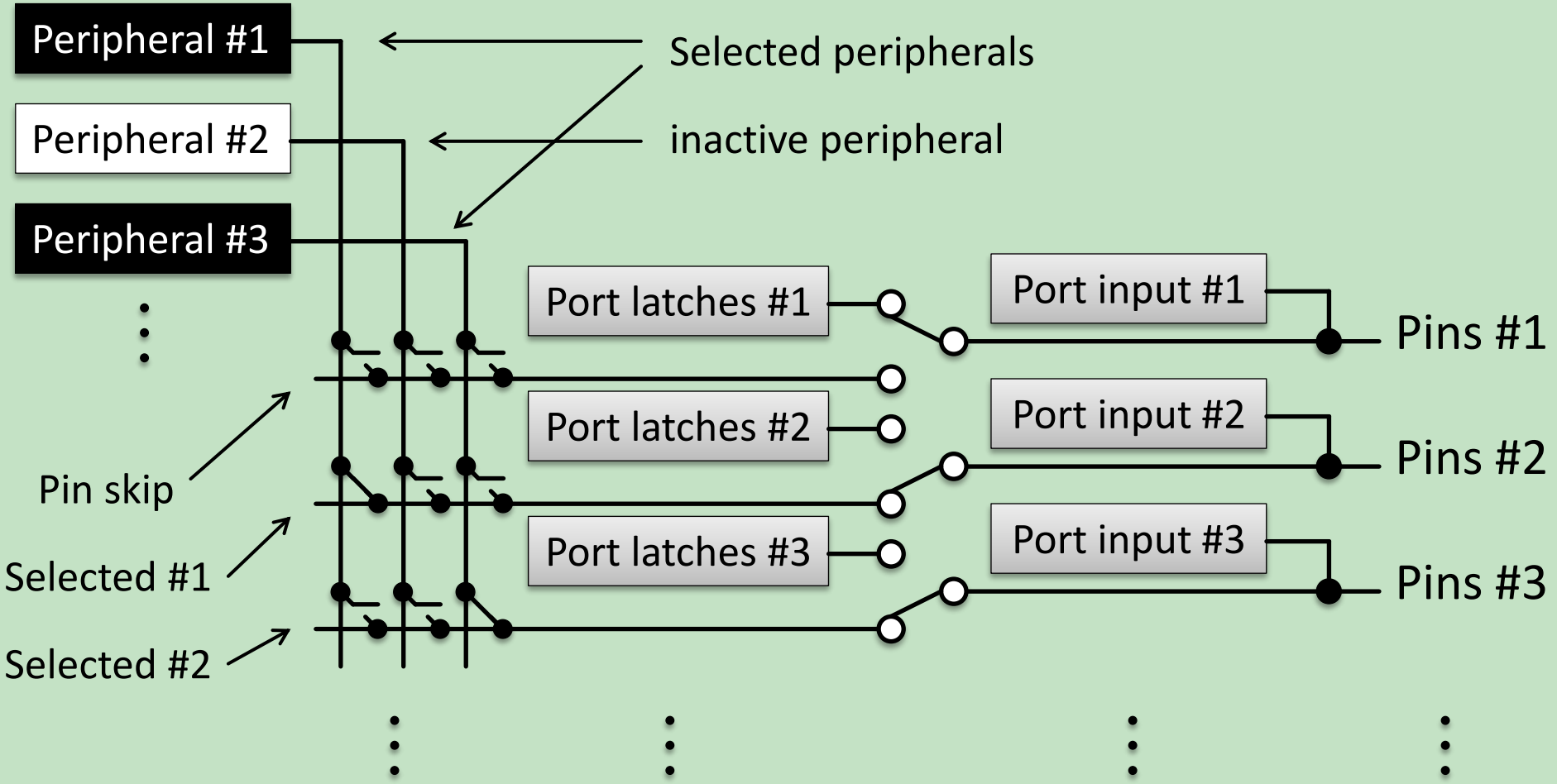
- A /r és /n kezelése nincs megoldva
  - nem nagyon van rá igény, inkább LCD_MoveTo
- A további LCD függvények szükségesek
  - clear, move, blink on/off, etc.

Crossbar

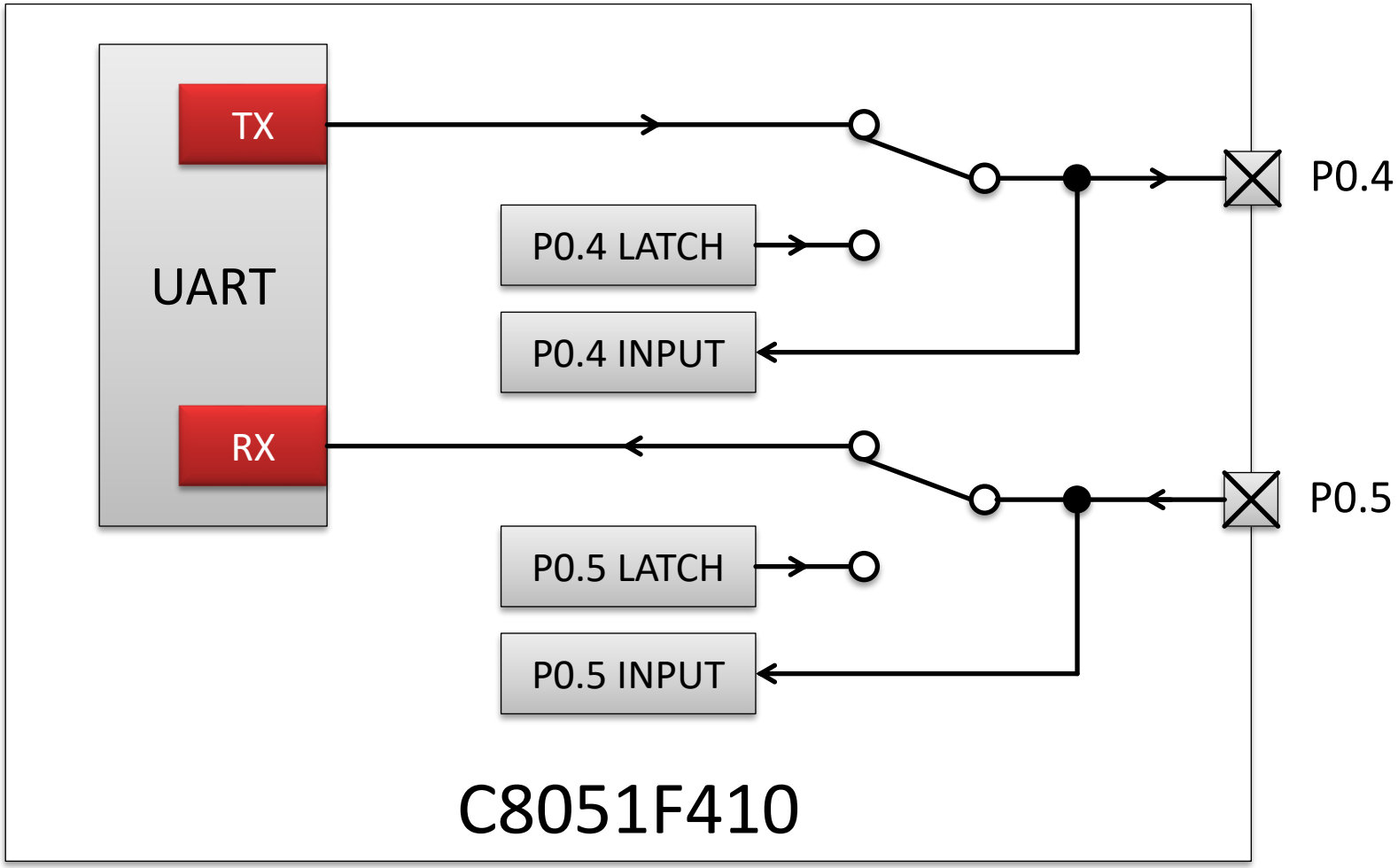
Crossbar – belső perifériák kivezetésekre kötése



Crossbar – belső perifériák kivezetésekre kötése



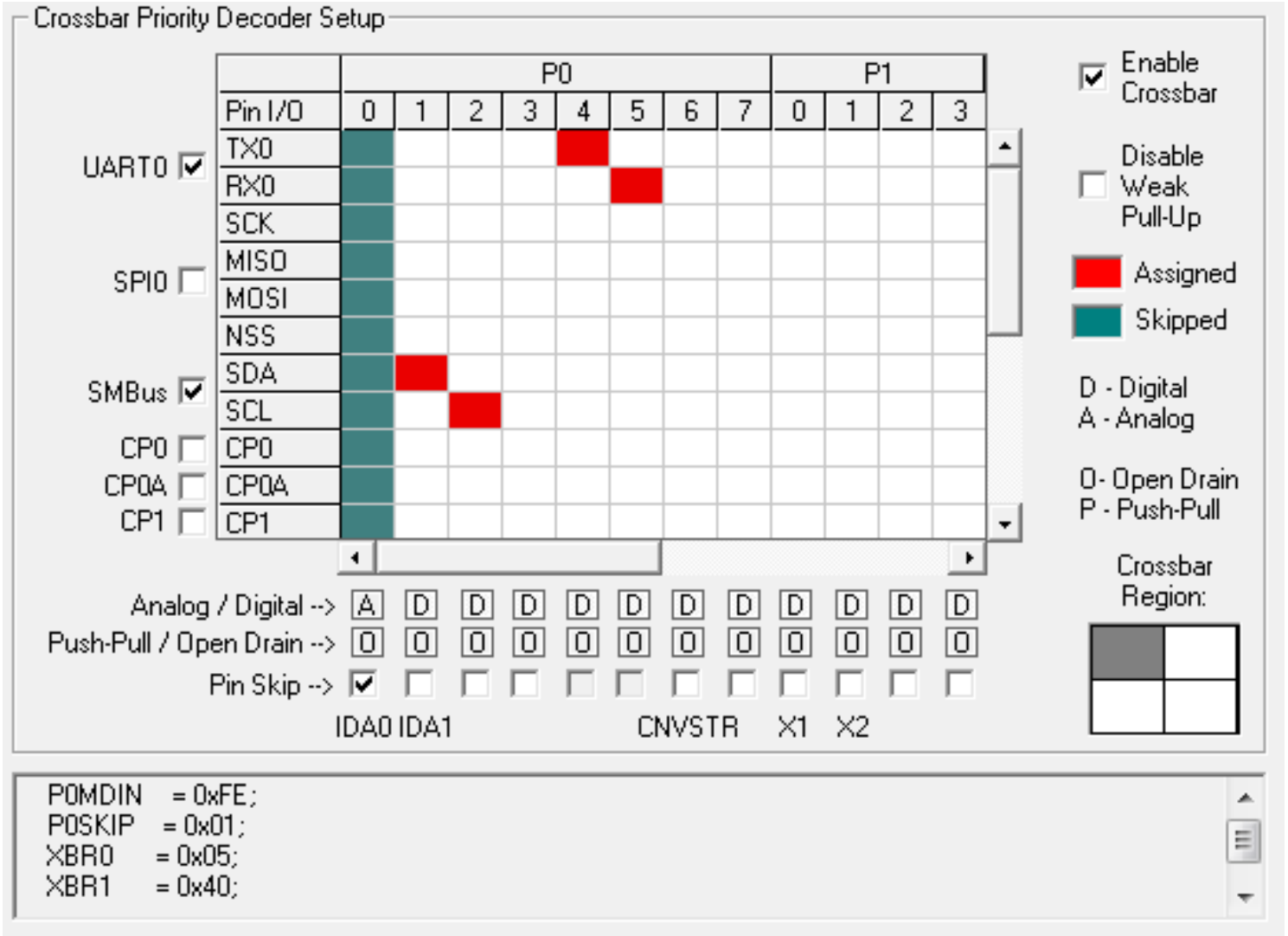
Crossbar – periféria és portok megosztása példa



Crossbar tulajdonságok

- A perifériák prioritási sorrendben kapcsolódnak a kivezetésekre (a portcella bitek sorrendjében)
- A bekapcsolt perifériák
  - foglalnak kivezetést
  - átveszik a kivezetés kezelését a processzortól
  - a társított kivezetés olvasható a processzor számára
- A maradék kivezetések normál portbitek
  - a processzor közvetlenül írhatja, olvashatja
- Lehetséges kivezetések kihagyása is (pin skip)
  - ekkor a crossbar a következő szabad kivezetést köti be

Crossbar ▸ példa: DAC0, SMBus, UART



Crossbar ▸ Megjegyzések

- Ha a crossbar nincs engedélyezve, a GPIO sincs!
  - Alapértelmezés reset után: crossbar kikapcsolt állapot
    - gyenge felhúzóellenállások be vannak kapcsolva
    - logikai bemenetként használhatók
    - külső meghajtás nélkül logikai 1 értéken vannak
- Ha a program futása közben változtatni kell a kiosztást:
  - a crossbar kikapcsolása
  - új kiosztás definiálása
  - a crossbar bekapcsolása
  - a külső jeleket ez befolyásolhatja!

Időzítő/számláló áramkörök

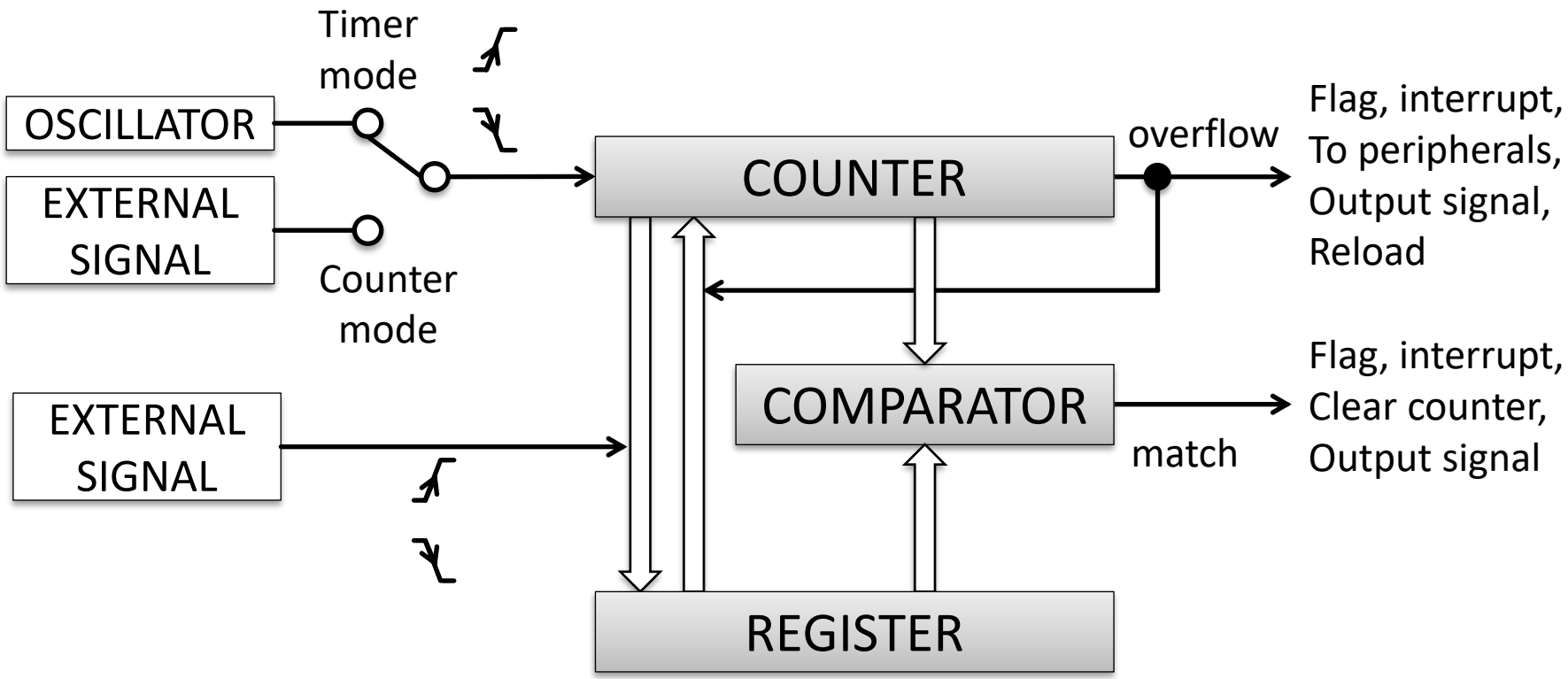
Időzítő/számláló áramkörök

- Időzítési/időmérési feladatok, például:
  - felhasználói felület kezelése
  - mechatronikai rendszerek
  - periodikus események (megszakítások)
  - mintavételezéses mérés, folyamatok figyelése
  - adatátvitel, kommunikáció ütemezése
  - időtartamok mérése
  - timeout generálás, késleltetések
  - időzített jelváltások, jelgenerálás
  - valós idejű óra

Időzítő/számláló áramkörök

- A fő elem: egy bináris számláló
  - Digitális bementi jel
  - A jel fel- vagy lefutó élének hatására eggyel növekszik vagy csökken az érték
- Időzítés:
  - Periodikus események számlálása
- Számlálás
  - Ismeretlen időpontokban bekövetkező események számlálása

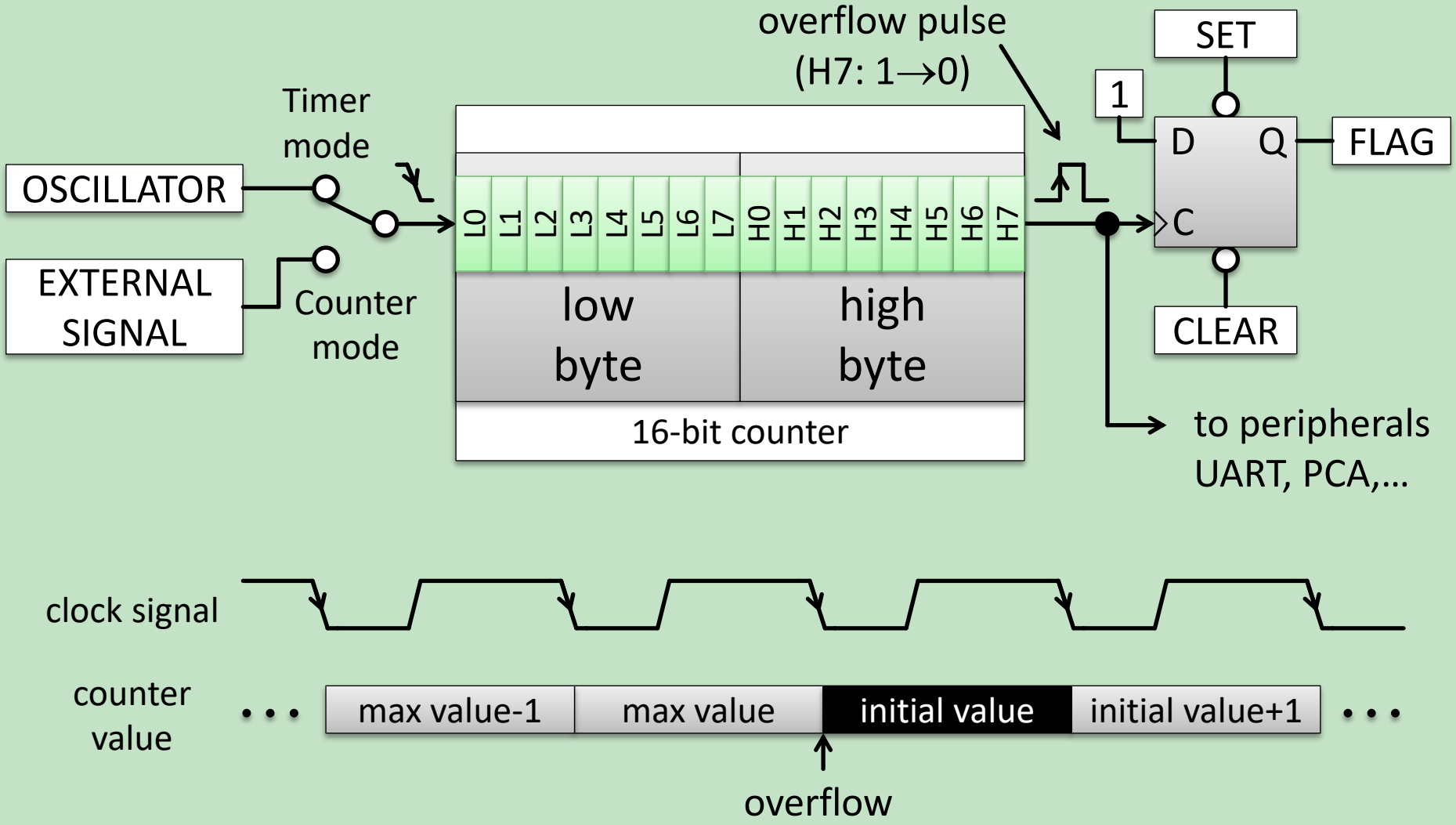
Hardver: számláló, regiszter, komparátor



Időzítő/számláló áramkörök

- Bemenete lehet ismert órajel vagy külső jel
- Eseményeket generálhat
  - Túlcsordulás
  - A regiszterrel megegyező érték
- A regiszterbe másolódhat érték vagy a regiszter tartalma másolódhat a számlálóba külső jel vagy szoftveres kérés hatására

Időzítő/számláló ▸ Működési elv



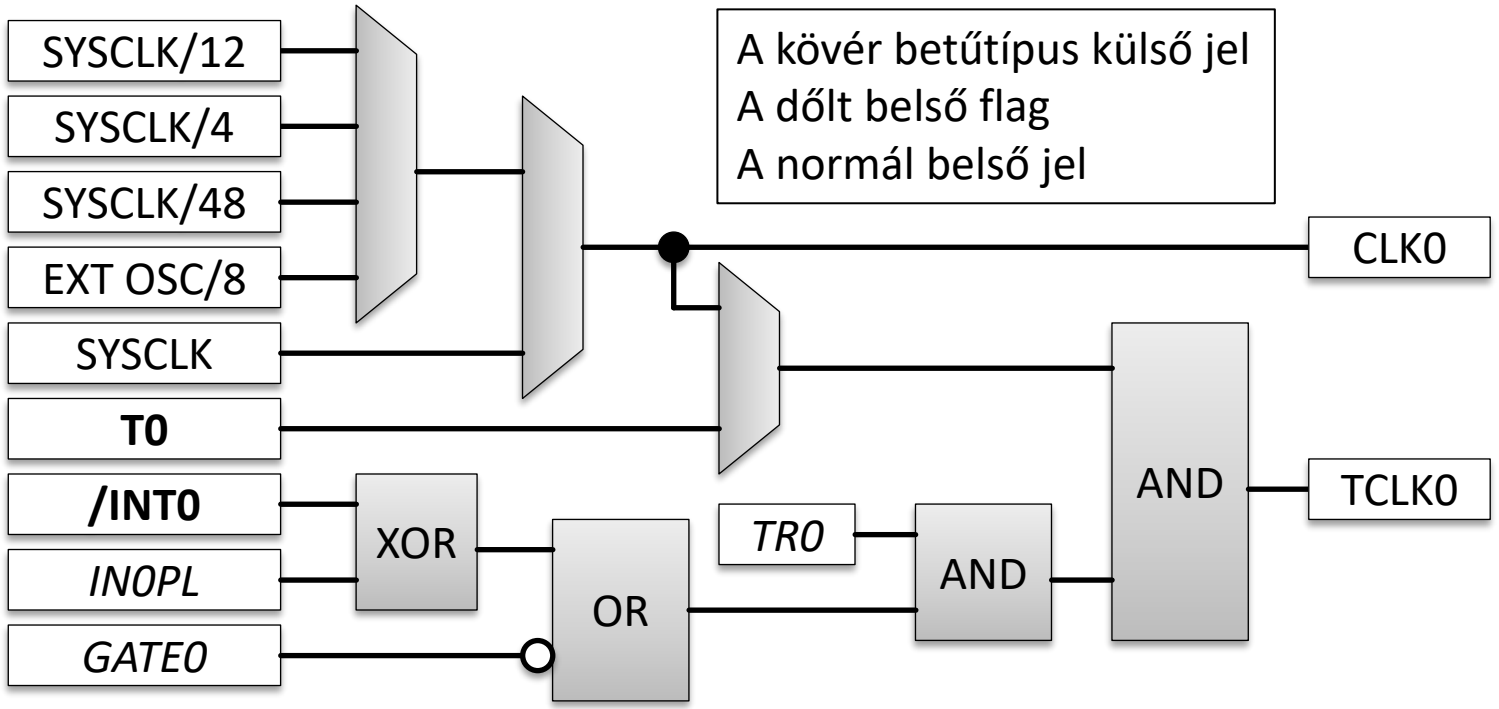
Időzítő/számláló ▸ Működési elv

- Bemenet: periodikus vagy külső jel
- Túlcsordulás
  - pulzust generál
  - beállít egy flaget
- A flag törlésig 1 értékű marad
- A pulzus meghajthat más perifériákat (*független a flagtől*)

Időzítő/számláló ▸ Timer0 és Timer1

- Klasszikus 8051 periféria
- Sokféle üzemmód
- Silicon Laboratories továbbfejlesztés
  - teljes órajelsebesség is (SYSCLK)
  - rugalmasabb megszakításkezelés

Időzítő/számláló ▸ Az órajel beállítása



- CLK0 az órajelforrásokból áll elő
- TR0 engedélyezi a számlálást
- GATE0 engedélyezi a külső jellel való kapuzást
- IN0PL invertálja a külső kapuzó jelet
- TCLK0 = (TIMER ? CLK0 :T0) & TR0 & (!GATE0 | (INT0 ^ IN0PL))
- Timer1 számára hasonló, de független beállítás

Timer0 és Timer1 ▸ Mode 0 és Mode 1

- 13-bites számláló
- Minden TCLK0 lefutóél egyet növel a számláló értékén
- A számláló két 8-bites részből áll
  - TH0: a nagyobb helyi értékű byte
  - TL0: a kisebb helyi értékű byte
- TL0-nak csak az alsó 5 bitje aktív, a felső három határozatlan
- TF0 1-re vált, ha a számláló túlcsordul
- TF0 megszakítást válthat ki, ha engedélyezve van, ekkor automatikusan törlődik, egyébként szoftveresen kell törölni
- **Mode 1 azonos, de mind a 16 bit aktív**

Timer0 és Timer1 ▸ Mode 0/1 alkalmazások

- Késleltetés generálás
- Megszakítás generálás
  - hardveres (fix időzítés)
  - szoftveresen felülírható
- Frekvenciamérés (VFC, R vagy C)
- Periódusmérés
- Időtartam mérése (gate használata)

Timer0 és Timer1 ▸ Késleltetés generálása

- A függvényhívás és műveletek ideje is hozzáadódik
- **`steps`** lépés idejéig várakozik

```
void Delay(unsigned short steps)
{
  TMOD=(TMOD & 0xF0) | 0x01; // 16-bit timer
  CKCON=CKCON | 0x04; // timer0 clk=sysclk
  TH0=-steps >> 8;    // 65536-steps
  TL0=-steps;
  TF0=0;
  TR0=1;
  while (!TF0);
  TR0=0;
}
```

Timer0 és Timer1 ▸ Megszakítás generálása

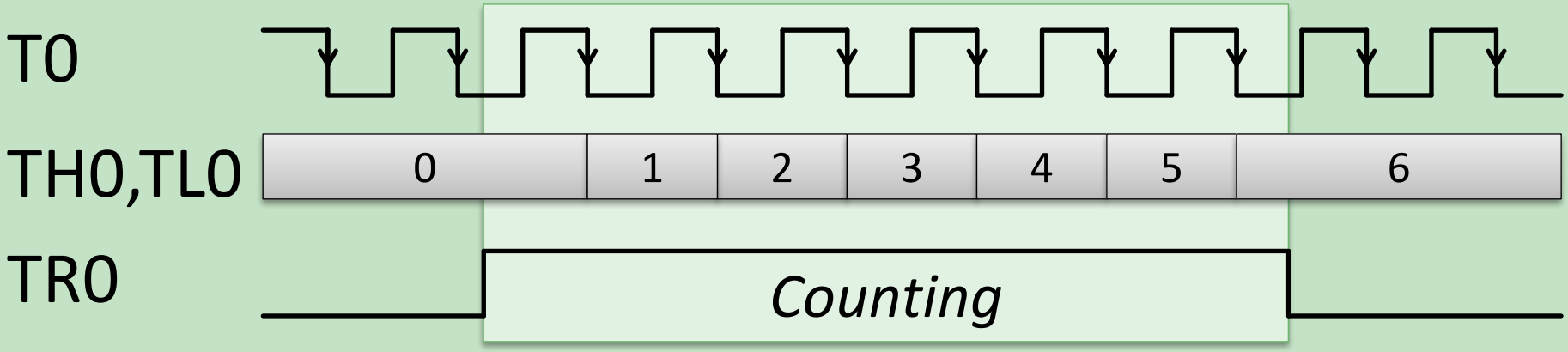
- RESET érték***: timer0 és 1*** órajel=sysclk/12
- `steps` timer lépésenként megszakítás (túlcsordulás)
- Megszakítási késleltetés befolyásolja, kicsit szoftverfüggő

```
TMOD=(TMOD & 0xF0) | 0x01; // 16-bit timer
TR0=1;     // run timer
IE=0x82;   // enable global & timer0 interrupts

void Timer0Handler(void) __interrupt 1
{
  TR0=0;              // stop timer
  TH0=-steps >> 8; // 65536-steps
  TL0=-steps;
  TR0=1;              // start timer
  …
}
```

Timer0 és Timer1 ▸ Frekvencia mérése

- Frekvencia, szenzorok, VFC, R vagy C mérése
- Adott ideig számláljuk a **lefutó éleket**
- Tartomány 0-fmax vagy fmin-fmax
- Felbontás?
- Két timer
  - az egyik számlálja az eseményeket
  - a másik az időtartamot (pontosság?)



Timer0 és Timer1 ▸ Frekvencia mérése

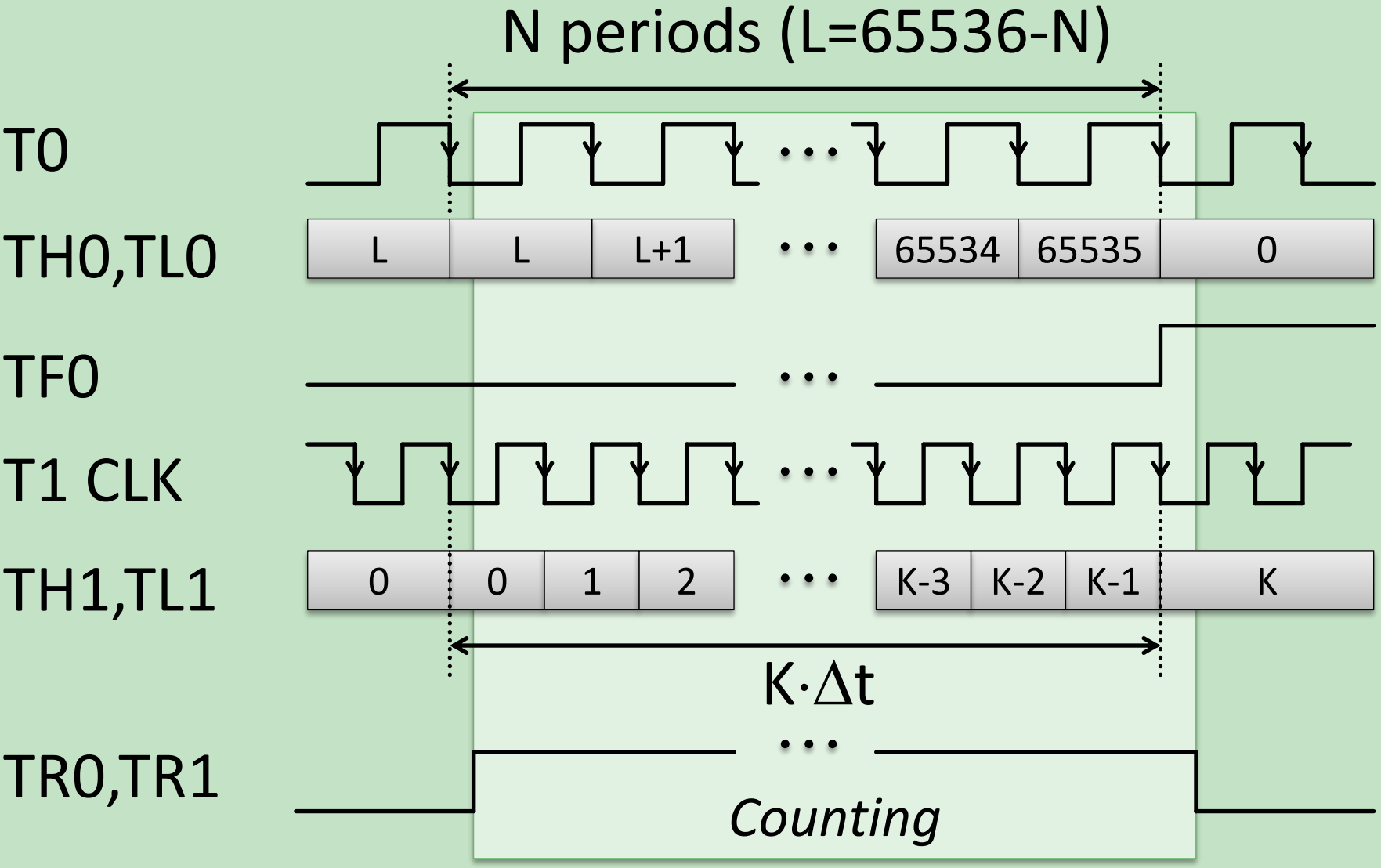
```
TCON=0;              // stop timers
TMOD=0x15;           // T0:16-bit counter T1:timer
TH0=0;               // initialise counter
TL0=0;               // initialise counter
TH1=-steps >> 8;     // 65536-steps
TL1=-steps;          // TR0=1 for steps*timer period
TF1=0;               // clear timer 1 flag
TCON=0x50;           // run both timers
while (!TF1);        // assembly?
TCON=0;              // stop both timers
```

- Az eredmény TH0/TL0 regiszterekbe kerül
- Esetleges megszakítások ronthatják az eredményt!
- A timerek másra ezalatt nem használhatók
- Összességében: kicsit **szoftverfüggő**, nem teljesen hardveres megoldás

Timer0 és Timer1 ▸ Periódus mérése

- N periódust megszámlálunk
- Megmérjük, mennyi T ideig tart
  - $T \approx K \cdot \Delta t$
  - Δt = timer lépésidő
- Periódus = $T/N \approx K \cdot \Delta t/N$
- Két timer használata
  - számlálás N esemény bekövetkeztéig
  - a szükséges időtartam megmérése, azaz K mérése

Timer0 és Timer1 ▸ Periódus mérése



Timer0 és Timer1 ▸ Periódus mérése

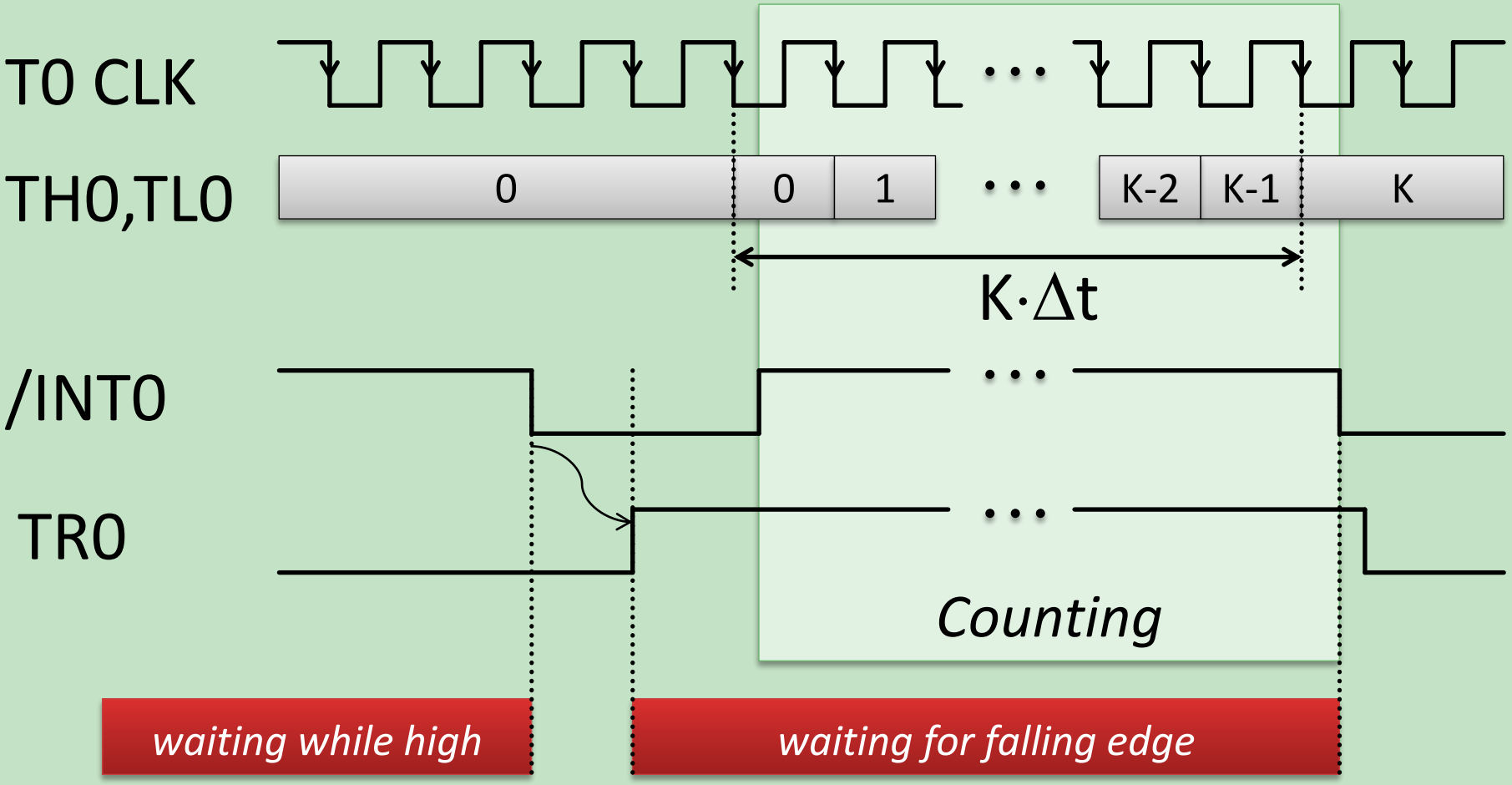
```
TCON=0;          // stop timers
TMOD=0x15;       // T0:16-bit counter T1:timer
TH1=0;           // clear T1 timer value
TL1=0;           // clear T1 timer value
TH0=-N >> 8;     // 65536-N
TL0=-N;          // N events to overflow (TF0=1)
TF0=0;           // clear timer 0 flag
TCON=0x50;       // run both timers
while (!TF0);    // wait for N events (assembly?)
TCON=0;          // stop both timers
```

- Az eredmény a TH1/TL1 regiszterekbe kerül (K)
- Esetleges megszakítások ronthatják az eredményt!
- A timerek másra ezalatt nem használhatók
- Összességében: kicsit **szoftverfüggő**, nem teljesen hardveres megoldás

Timer0 és Timer1 ▸ Időtartam mérése

- Digitalizálás timer periódusok számlálásával
- A timerek **GATE** funkciója használható
- Megoldás GATE funkcióval:
  - A számláló adott órajellel megy
  - A külső jel aktív idejéig engedi a GATE a számlálást
  - Elég egy timer használata
  - INT0 engedélyezése a crossbaron szükséges, ez a GATE-el engedélyezhető bemenet egyben

Timer0 és Timer1 ▸ Időtartam mérése

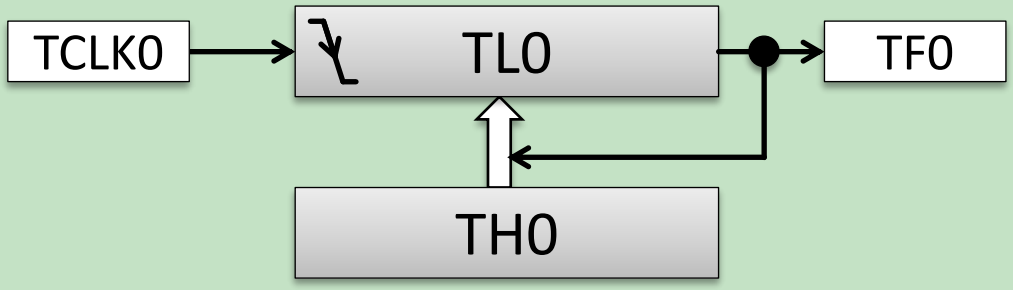


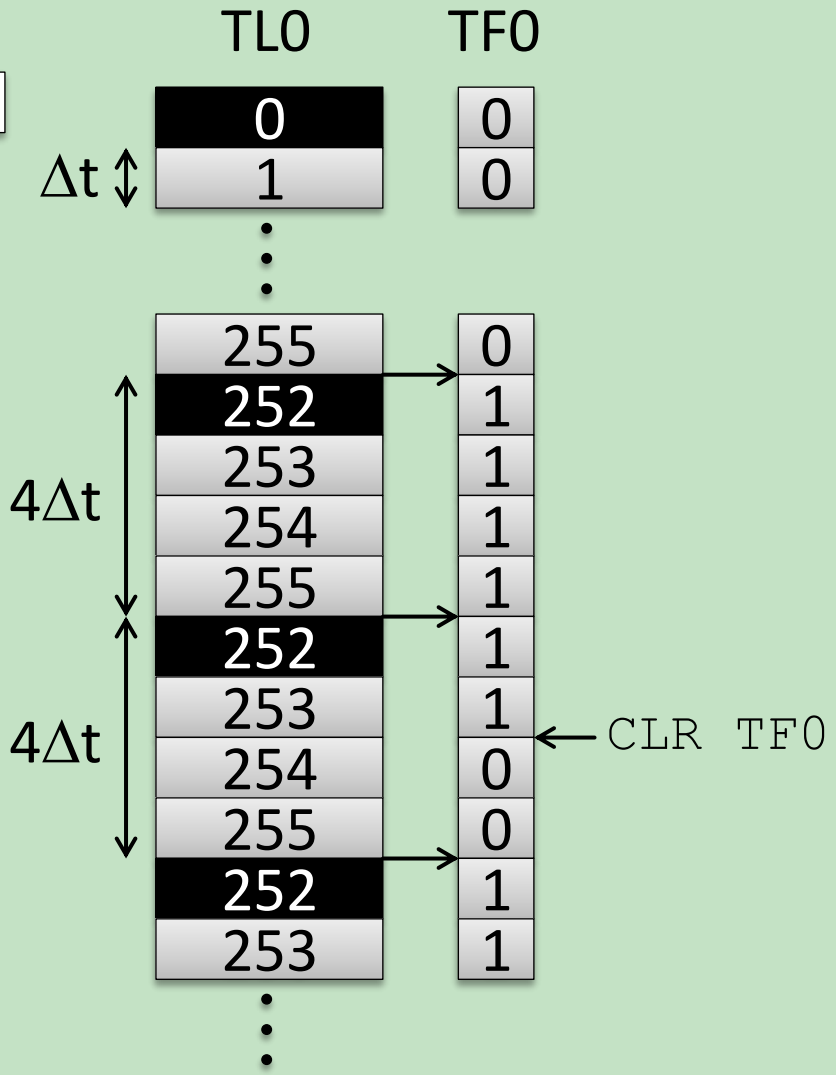
Timer0 és Timer1 ▸ Időtartam mérése

```
TR0=0;          // stop timer
TH0=TL0=0       // clear timer
TMOD=0x09;      // T0:16-bit gated timer mode
IT0=0;          // level triggered /INT0
IE0=0;          // clear INT0 flag
while (!IE0);   // wait for input going down
IT0=1;          // edge triggered /INT0
IE0=0;          // clear INT0 flag
TR0=1;          // enable timer
while (!IE0);   // wait for end of pulse
TR0=0;          // stop timer
```

- Az eredmény a TH0/TL0 regiszterekbe kerül
- A pulzusnak TR0=1 után kell indulnia! Erre várakozás a kódban
- INT0-t a crossbaron engedélyezni kell, megszakításban is kezelhető
- Esetleges megszakítások ronthatják az eredményt!
- Kicsit **szoftverfüggő**, nem teljesen hardveres megoldás

Timer0 és Timer1 ▸ Mode 2: Auto reload





- 8-bites számláló
- TH0 tartalmazza a kezdőértéket
- Ez TL0 túlcsordulásakor beíródik
- innen folytatódik a számlálás
- Példa: TL0=0; TH0=252; TF0=0;
  - **Periódus: (256-TH0)·Δt**
- TF0:
  - szoftver vagy megszakítás törli

Timer0 és Timer1 ▸ Mode 2 alkalmazások

- Soros bitátviteli ráta: baud rate (UART)
- Periodikus megszakítások generálása
- Kiszolgáló a PCA-hoz (Timer0, lásd később)
- 8-bit: korlátozott felbontás és pontosság
- Nagyobb periódus – nagyobb pontosság
- Mindig ellenőrizzük, mennyi lett a periódus

Timer0 és Timer1 ▸ Mode 2 alkalmazások

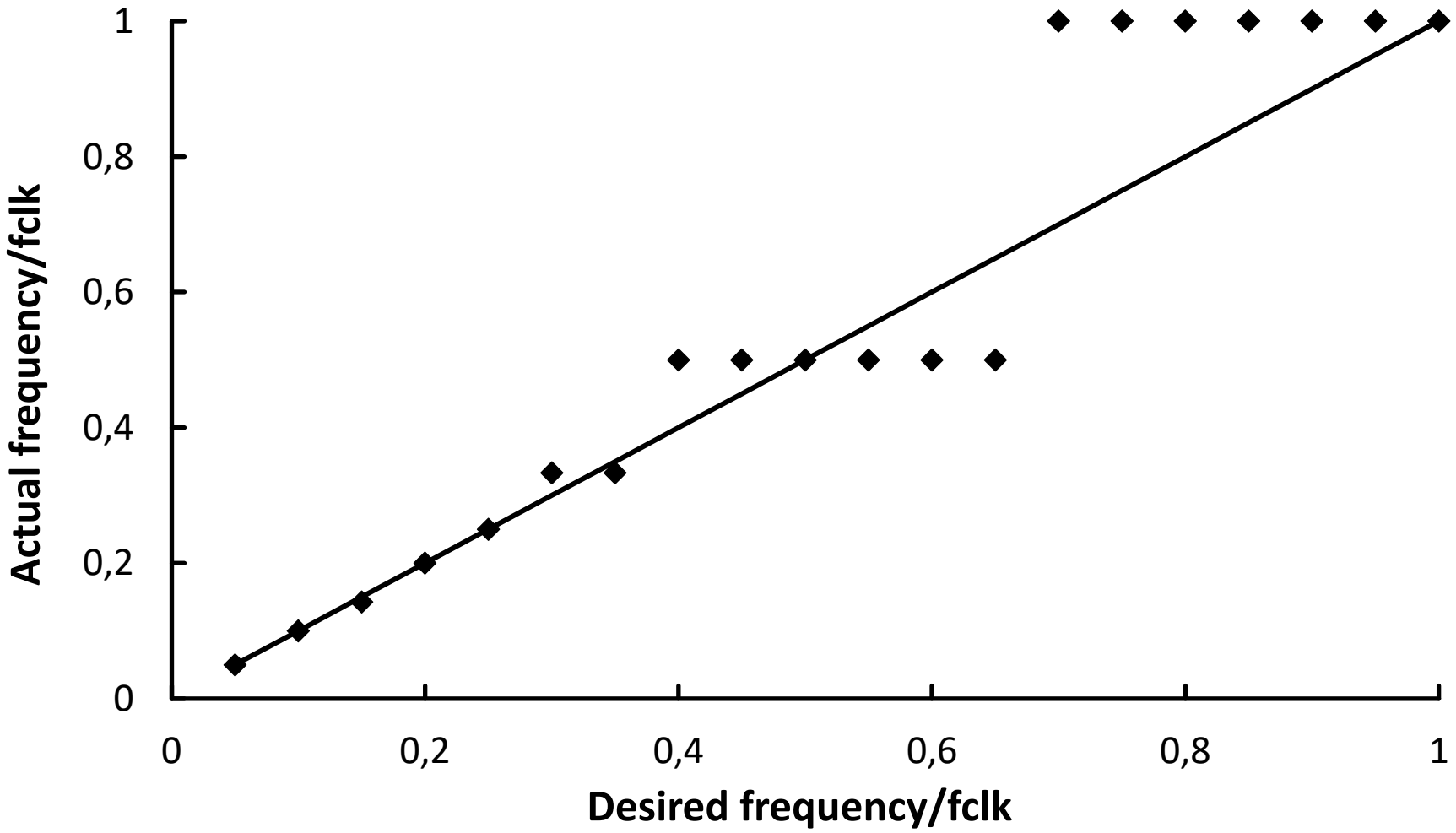
$$Period \approx \left(256 - TH0\right) \cdot \Delta t = \left(256 - TH0\right) / \ f_{clk}$$

$$1/\ Period = \ f_{desired} \ \approx \ f_{actual} \ = \ f_{clk} \ / \left(256 - TH0\right)$$

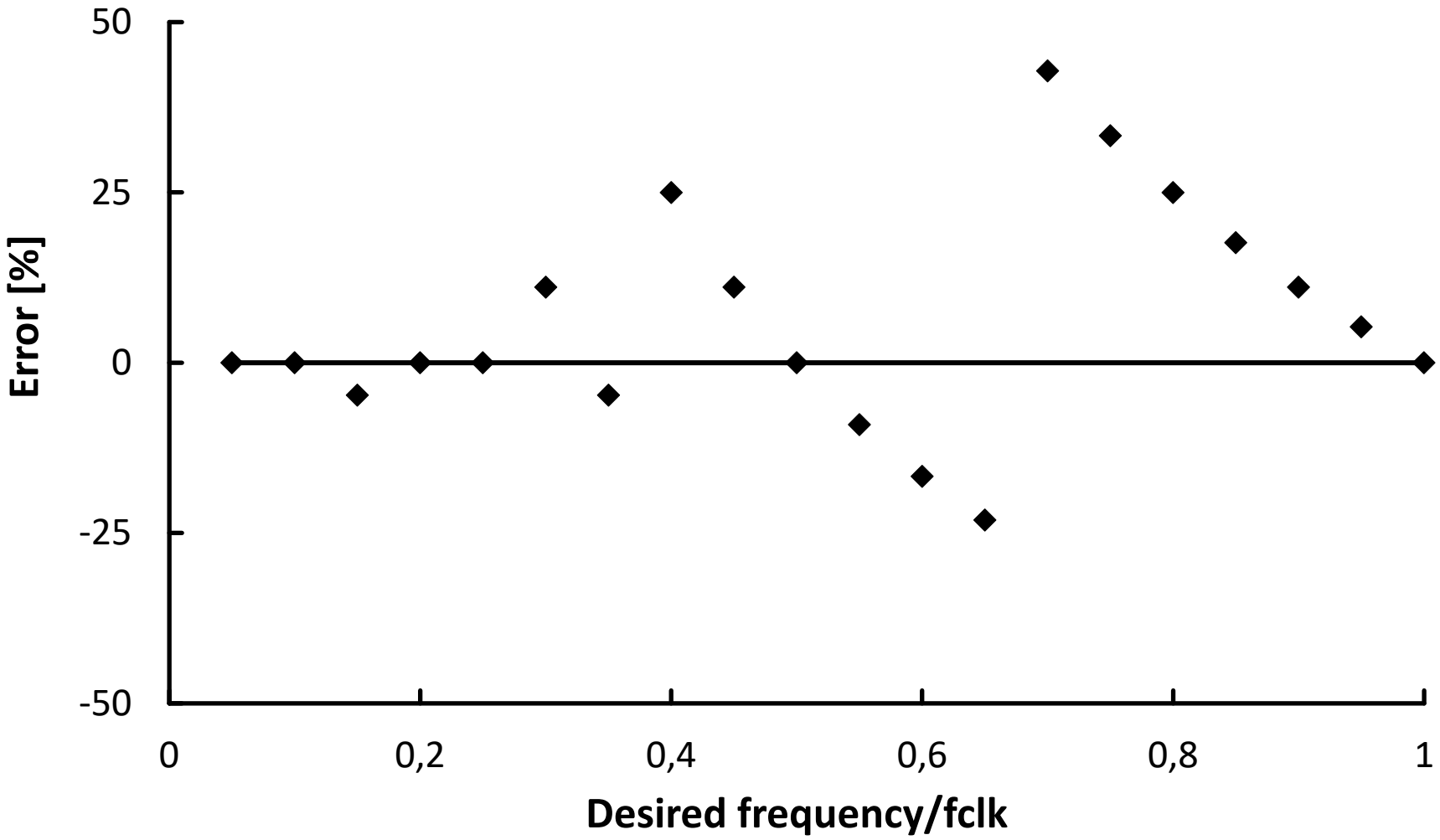
$$TH0 = \left\lfloor 256 - \ f_{clk} \ / \ f_{desired} \ + 0.5 \right\rfloor$$

$$f_{actual} = \ f_{clk} \ / \left(256 - \left\lfloor 256 - \ f_{clk} \ / \ f_{desired} \ + 0.5 \right\rfloor\right)$$

Timer0 és Timer1 ▸ Mode 2 alkalmazások



Timer0 és Timer1 ▸ Mode 2 alkalmazások



Timer0 és Timer1 ▸ Baud rate

- Baud rate = Timer1 overflow rate / 2
- F410 SYSCLK: 191406Hz
- **9600** bit/s = 191406/(256-TH1) / 2
- TH1=256-SYSCLK/BAUDRATE/2≈246=0xF6
- BAUDRATE=**9570** bit/s, jó ez? SYSCLK: 2%!

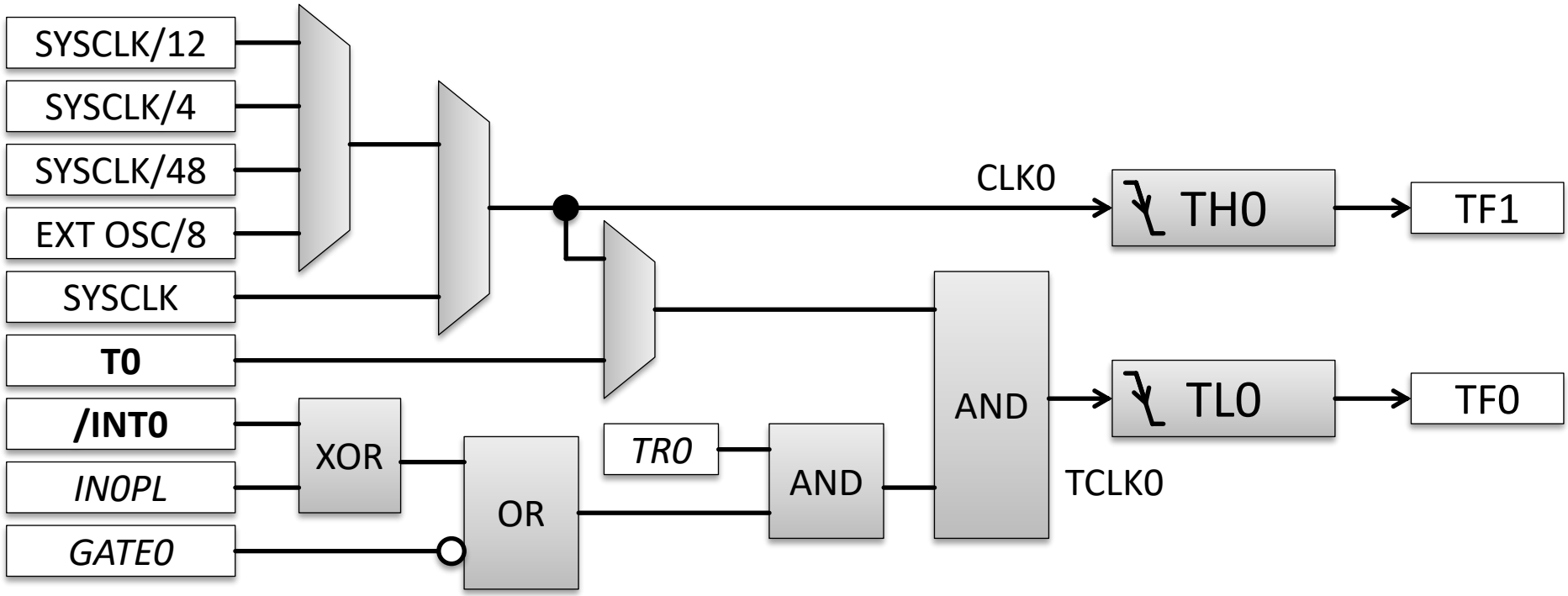
```
TMOD=(TMOD & 0x0F) | 0x20; // mode 2
CKCON=CKCON | 0x08;        // timer1clk=sysclk
TH1=0xF6;
TR1=1;
```

Timer0 és Timer1 ▸ Periodikus megszakítások

- Interrupt rate = Timer0 overflow rate
- F410 SYSCLK: 191406Hz
- TH0=256-SYSCLK/IRQRATE
- Elvi tartomány: 748Hz-191406Hz (TH0=0-255)
- A periódus > IRQ végrehajtási idő legyen

```
TMOD=(TMOD & 0xF0) | 0x02; // mode 2
CKCON=CKCON | 0x04;        // timer0clk=sysclk
TH0=65;                    // 1000Hz
TR0=1;
IE=0x82;                   // enable interrupt
```

Timer0 és Timer1 ▸ Mode 3



- Két független 8-bites számláló (Timer 1 inaktív ekkor)
- TL0-t a szokásos komplex órajel hajtja
- TH0-t a skálázott órajel vagy SYSCLK

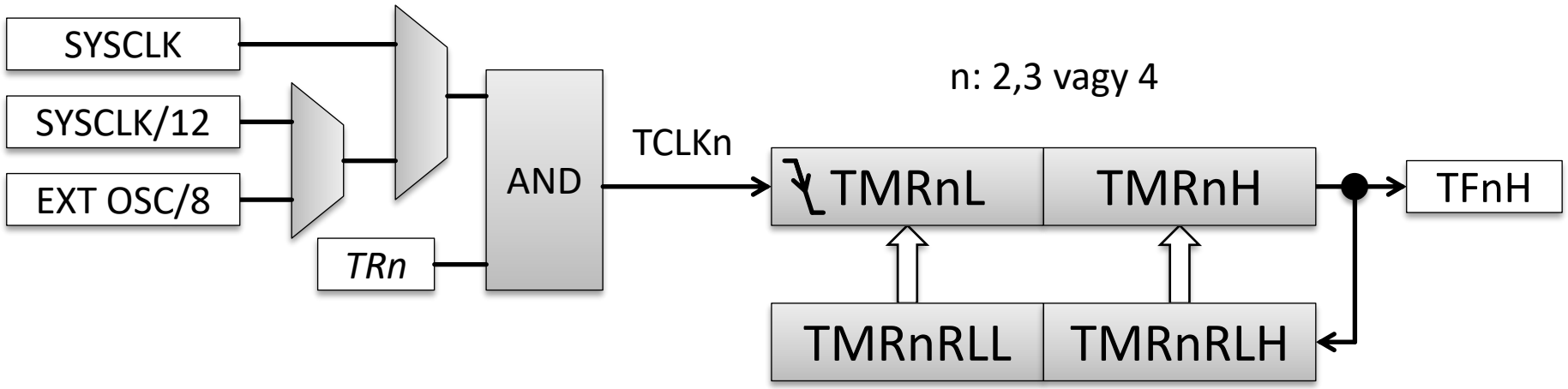
Timer0 és Timer1 ▸ Mode 3 ▸ Alkalmazások

- Periodikus megszakítások generálása
- TL0 számlálásra, megszakítások generálására
- Órajel a perifériák számára

Időzítő/számláló ▸ Timer2-Timer4

- Az MCS-51 mikrovezérlőben: Timer 0 és 1
- 8052: Timer 2 hozzáadása
- Silicon Labs
  - további timerek hozzáadása
  - 16-bit auto reload – nagyobb tartomány, pontosság

Időzítő/számláló ▸ Timer2-Timer4 ▸ Auto reload



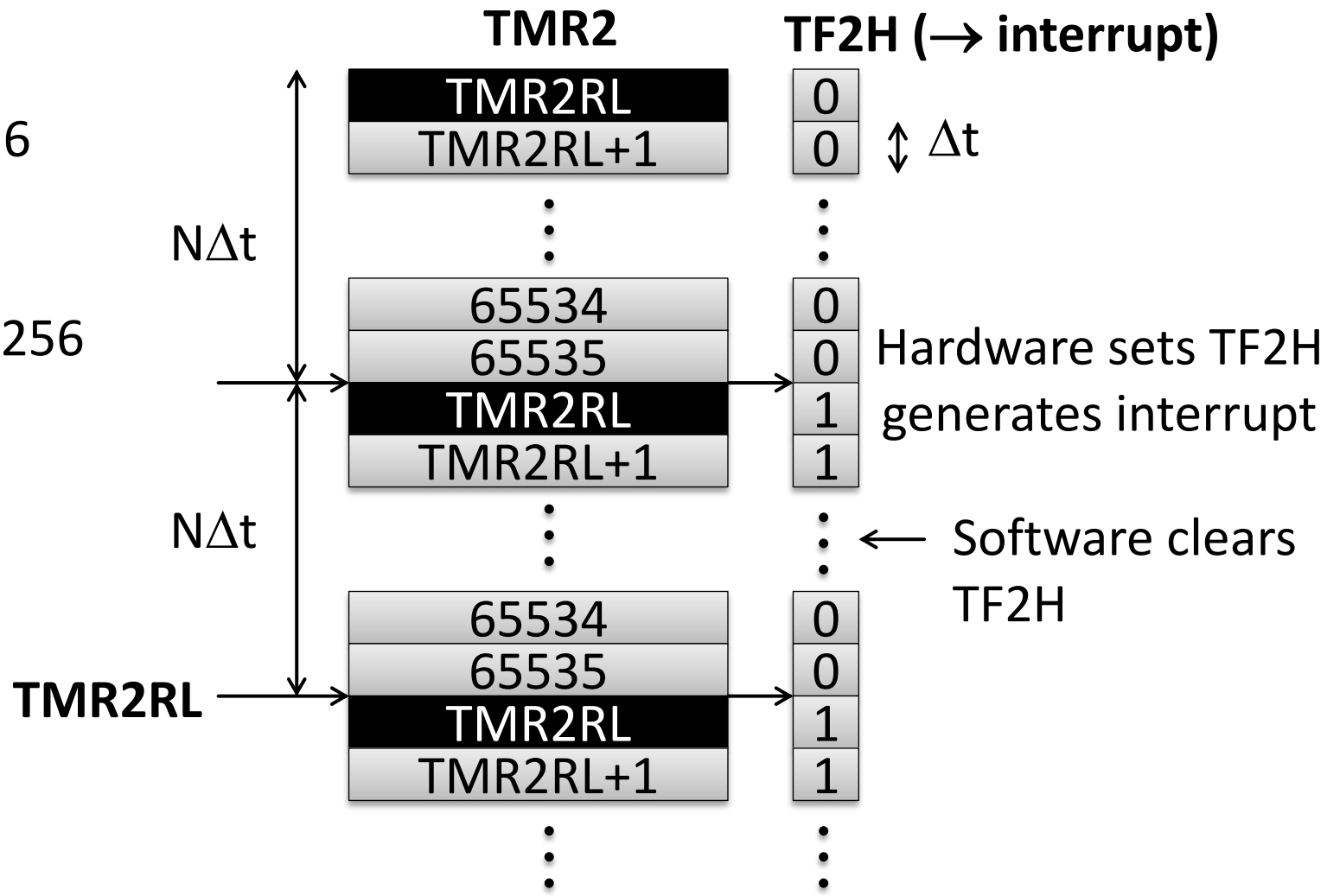
- C8051F410

Időzítő/számláló ▸ Timer2-Timer4 ▸ Auto reload

- TMRnRL: 16-bites érték
- A túlcsordulási periódus
  - (65536-TMRnRL)·Δt
  - Δt a TCLKn periódusideje
- A tulcsordulási frekvencia:
  - f_{TCLKn}/ (65536-TMRnRL)

Időzítő/számláló ▸ Timer2-Timer4 ▸ Auto reload

TMR2
=TMR2H*256
+TMR2L
TMR2RL
=TMR2RLH*256
+TMR2RLL



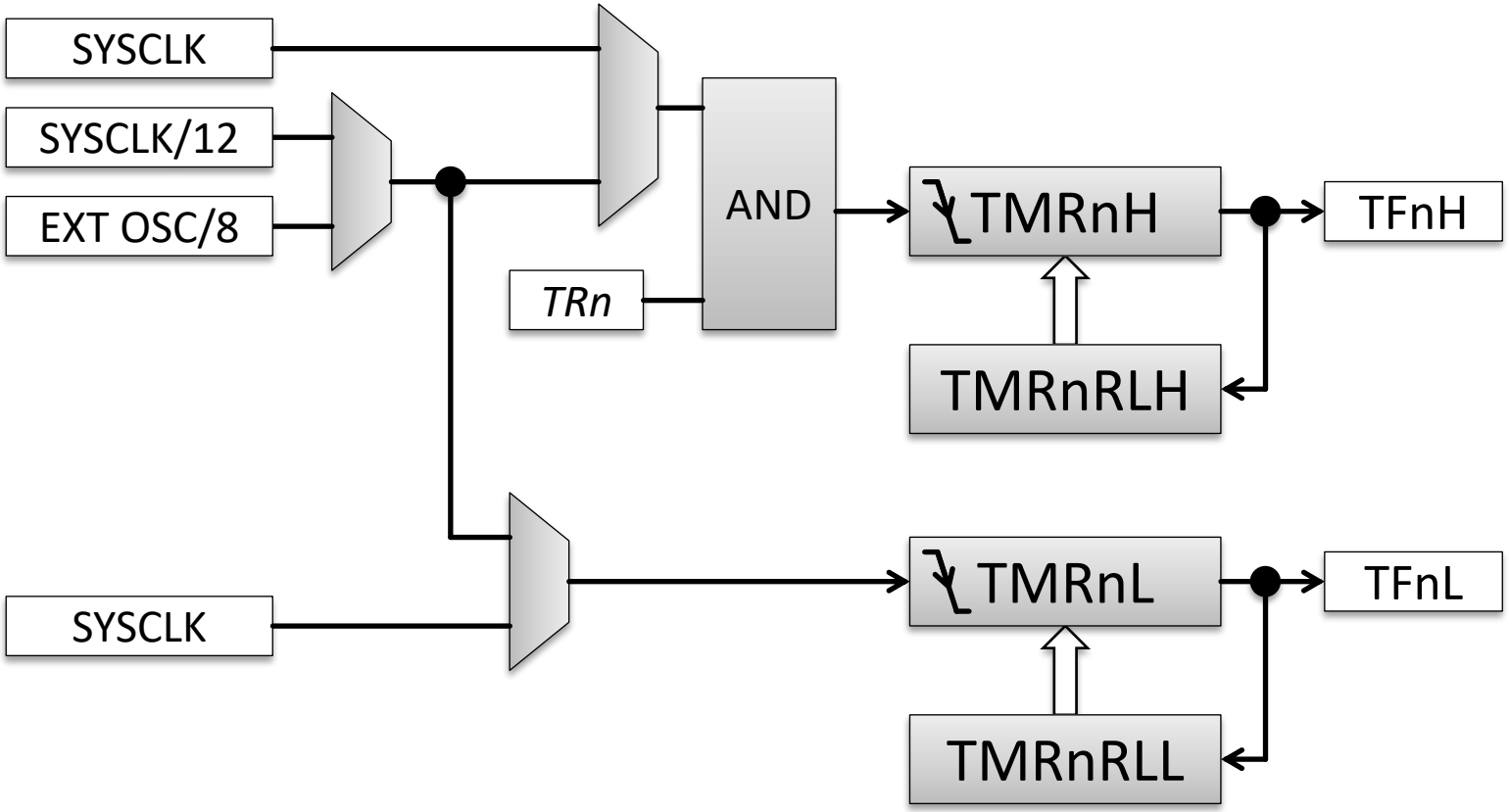
Időzítő/számláló ▸ Timer2-Timer4 ▸ Auto reload

```
/*************************
steps = period/dt
dt=1/timer clock
steps = timer clock*period
reload value = 65536-steps
*************************/

unsigned long period;  // in us
unsigned long tmrclk;  // timer clock in Hz
unsigned short tmrrl;  // reload value

tmrrl = -tmrclk*period/1000000L;
```

Időzítő/számláló ▸ Timer2-Timer4 ▸ Auto reload



Időzítő/számláló ▸ Timer2-Timer4 ▸ Auto reload alkalmazások

- Flag felhasználásával (megszakítás engedélyezés):
  - Periodikus megszakítások generálása
- A túlcsordulási pulzussal (***timer megszakítás nincs!***):
  - Baud rate generátor (UART)
  - SMBus bit rate, timeout
  - A/D konverzió indítása, mintavételezéses mérés
  - D/A konverzió indítása, időfüggő jel generálása
- Code profiling
  - SYSCLK legyen a számláló órajele
  - a kódrészlet elején a számláló nullázása és indítása
  - a kódrészlet végén a számláló állása a futási időt adja

Időzítő/számláló ▸ LED villogtatása

```
$include (C8051F410.INC)

LED     EQU  P0.2

CSEG at 0000h
   jmp  Main    ; reset, jump to the main

ORG 002Bh               ; Timer 2 IRQ vector
   anl TMR2CN,#07Fh ; clear interrupt flag
   cpl LED              ; complement LED
   reti                 ; return from interrupt
```

Időzítő/számláló ▸ LED villogtatása

```
Main:
    anl   PCA0MD,   #0BFh   ; watchdog off
    mov   PCA0MD,   #000h
    mov   XBR1,     #040h   ; crossbar on
    mov   TMR2RLL,  #0B2h   ; Timer 2 reload register
    mov   TMR2RLH,  #0C1h   ;
    mov   TMR2L,    #0B2h   ; Timer 2 counter initial value
    mov   TMR2H,    #0C1h
    mov   TMR2CN,   #004h   ; Start Timer 2
    mov   IE,       #0A0h   ; enable interrupts
    jmp   $                 ; repeat forever

END
```

fclk=1/Δt=191406/12 Hz ≈ 15950 Hz (RESET utáni alapérték)

1 sec = 15950·Δt → TMR2RL = 65536-15950 = 49586 = **0xC1B2**

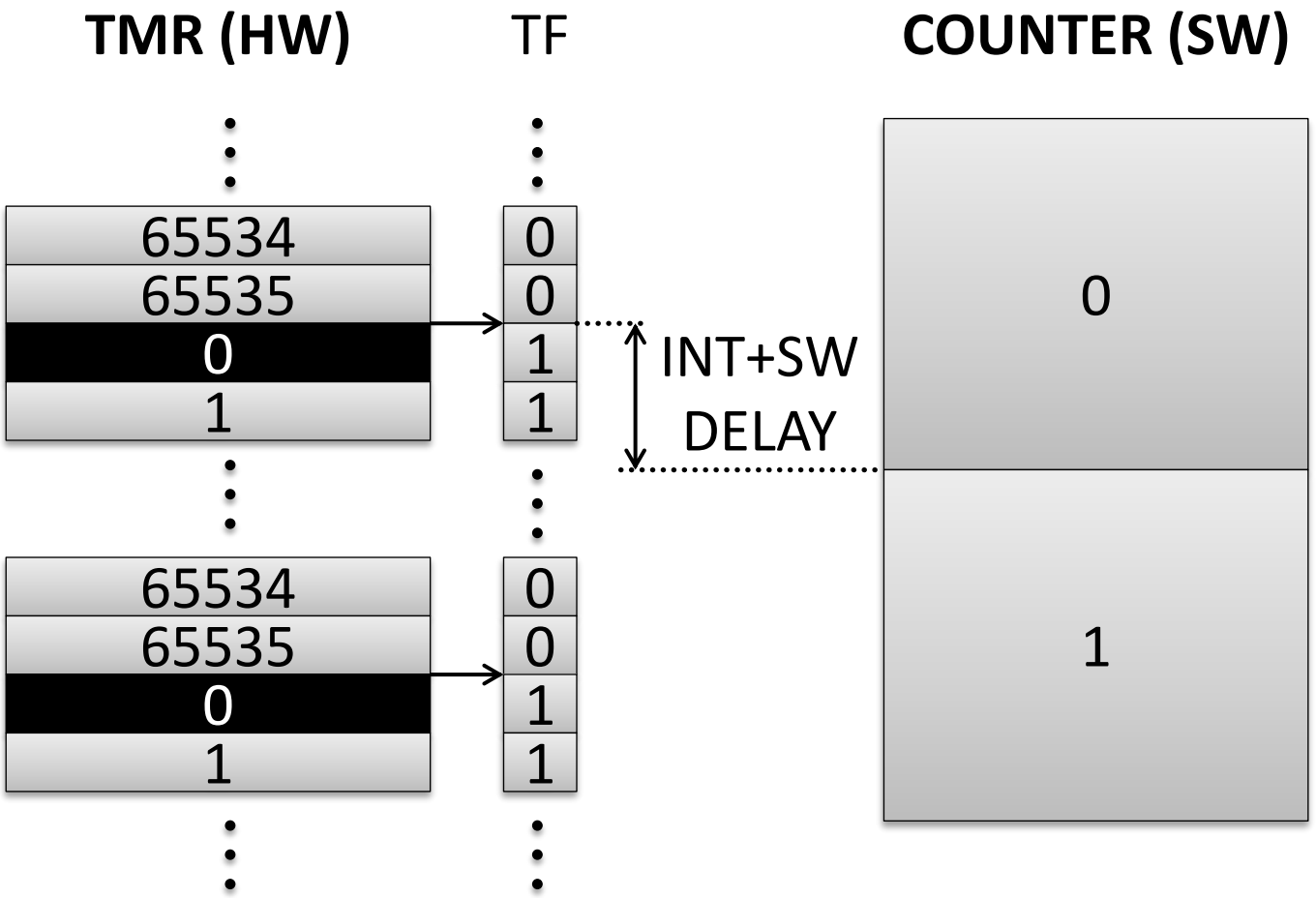
Időzítő/számláló ▸ Timer2-Timer4 ▸ >16-bit?

- Kevés a 16-bit?
- ***Szoftveres*** kiterjesztés:
  - Túlcsorduláskor egy változó értékét növeljük
  - Megszakítási rutinban

```
void TimerIRQ(void) __interrupt TIMER_VECTOR
{
   static unsigned char counter=0;

   counter = (counter+1) % countermax;
   if (!counter) Process();  // overflow
}
```

Időzítő/számláló ▸ Timer2-Timer4 ▸ >16-bit?

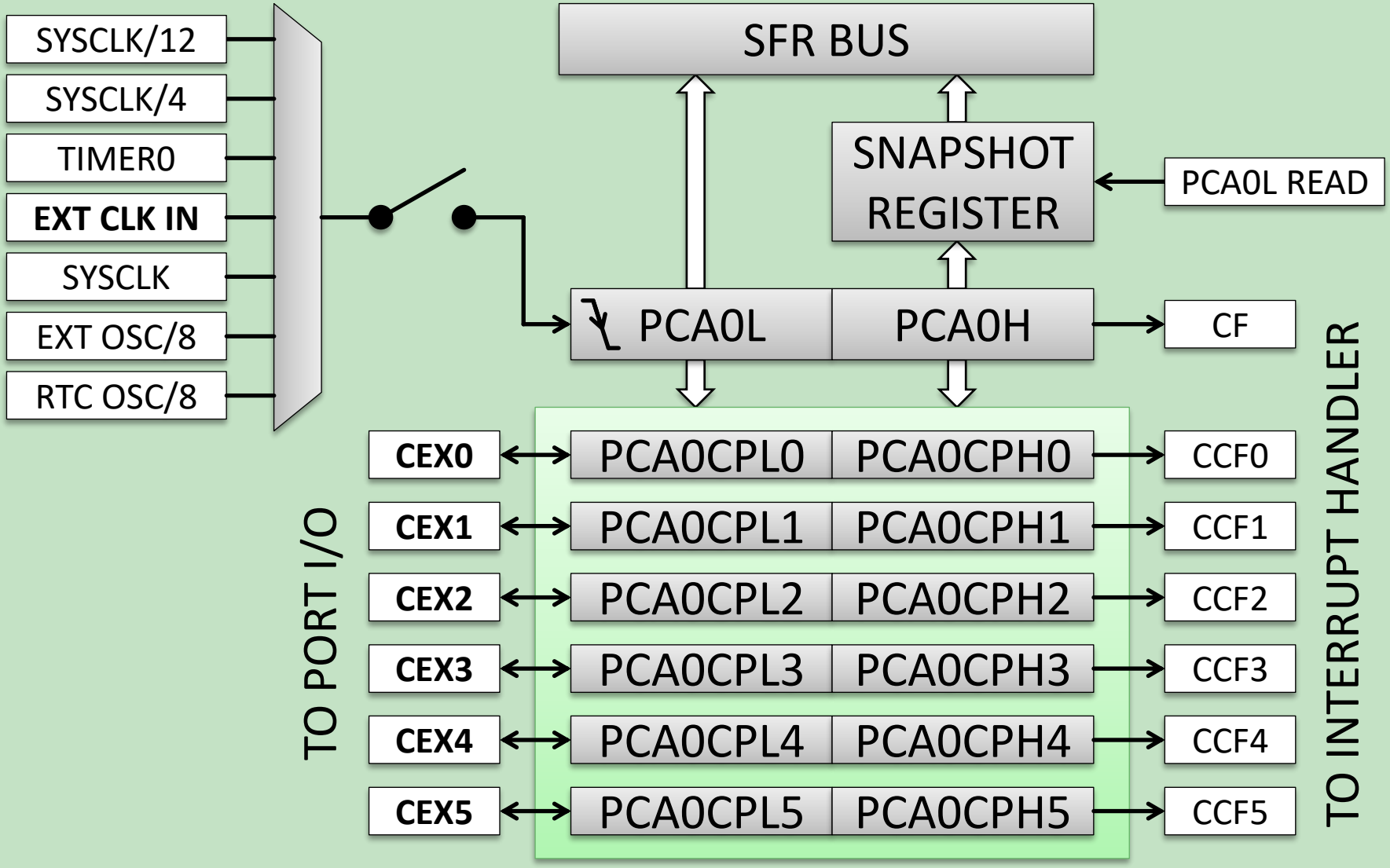


Programozható számlálótömb

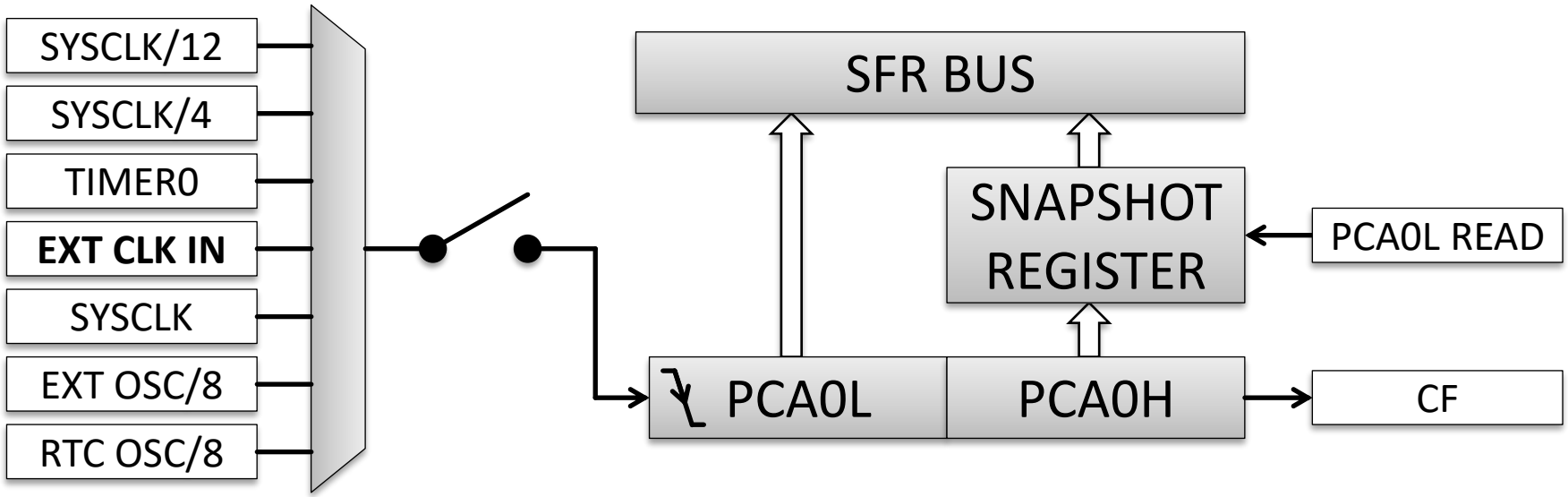
PCA: Programmable Counter Array

- Adott órajellel fut egy 16-bites számláló
- Mellette van több 16-bites regiszter
- Események
  - A számláló túlcsordul
  - A számláló értéke azonos egy regiszterével
  - Külső jel változása (felfutás, lefutás vagy mindkettő)
- Lehetséges hatások
  - Megszakítás generálása
  - A számláló értéke átmásolódik egy regiszterbe
  - Kimeneti jel változtatása

PCA ▸ A fő számláló meghajtása

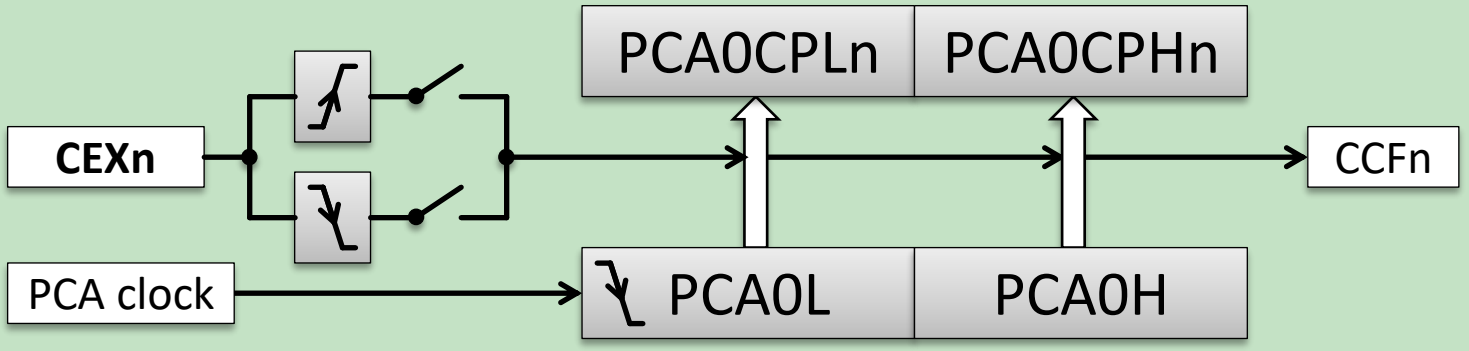


PCA ▸ A fő számláló meghajtása



- Sokféle órajelforrás
- Akár Timer0 8-bit auto-reload túlcsordulás
  - Timer 0 16-bittel való kiterjesztése
  - Lassabb, programozható órajel
- 16-bites biztonságos olvasás (PCA0L először)

PCA ▸ Edge-triggered capture mode

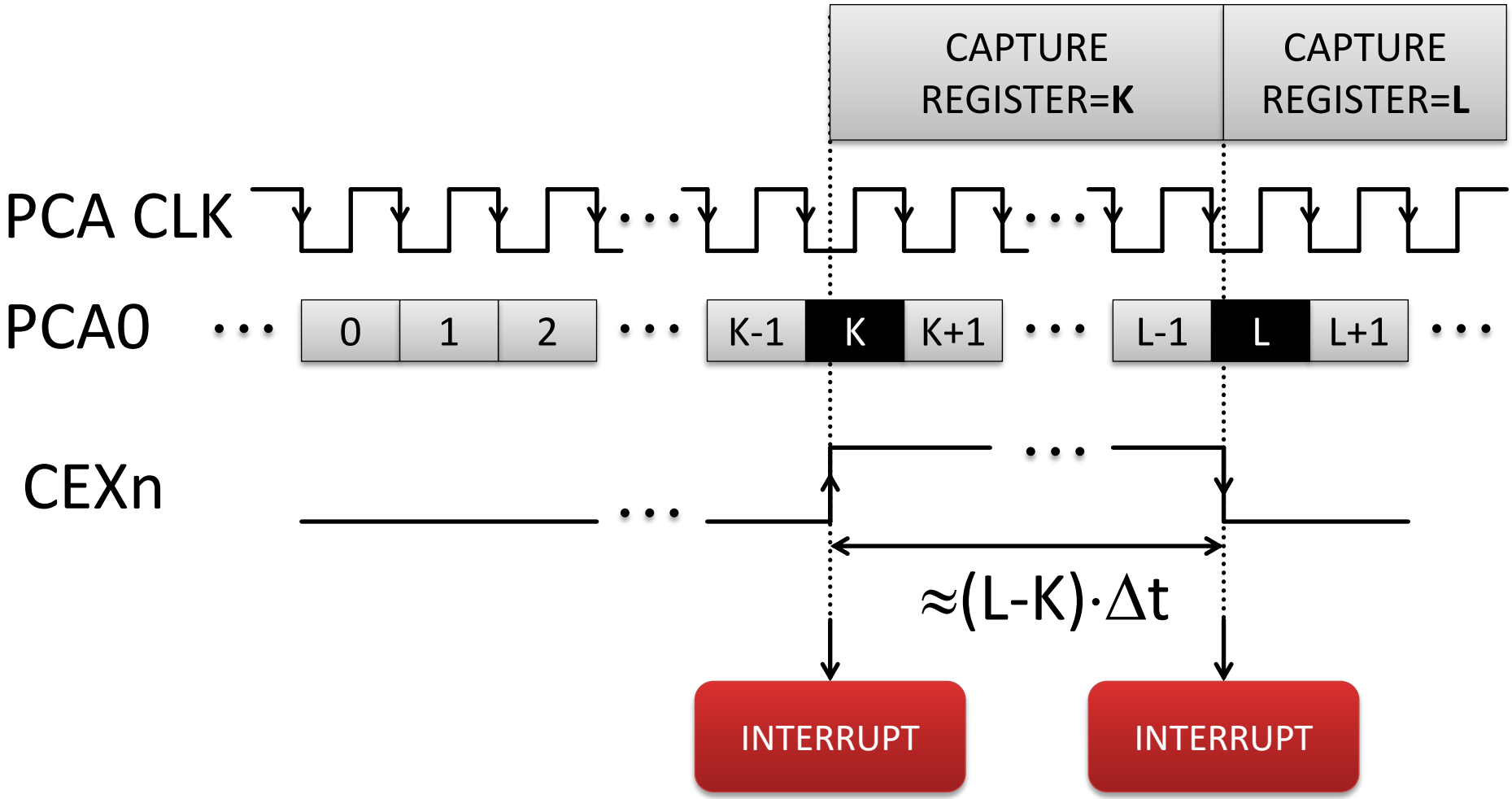


- **Külső jel** felfutó, lefutó vagy mindkét átmeneténél
- A PCA0 számláló a PCACPn regiszterekbe kerül
- A CCFn flag 1-re vált, megszakítás is létrejöhet
- Események, jelváltások időpillanatainak detektálására
- Frekvencia, periódusidő, pulzusszélesség, kitöltési tényező mérésére alkalmas

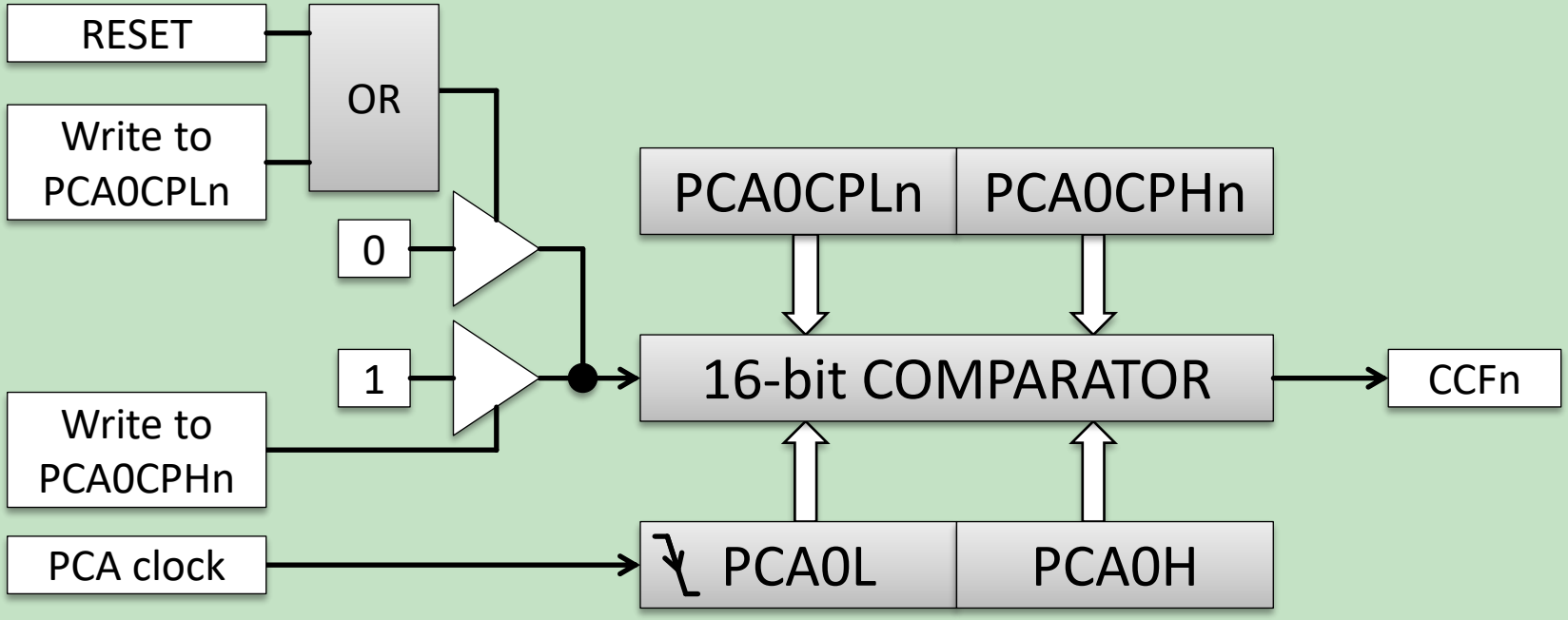
PCA ▸ Edge-triggered capture ▸ Példa

- Pulzusszélesség mérése
  - Felfutó él engedélyezése
  - Megszakítás engedélyezése
  - A megszakításban:
    - capture regiszter elmentése
    - lefutó él engedélyezése
    - következő megszakításban
      - capture regiszter aktuális és előző értékének különbsége
- Házi feladat: a kód megírása

PCA ▸ Edge-triggered capture ▸ Példa

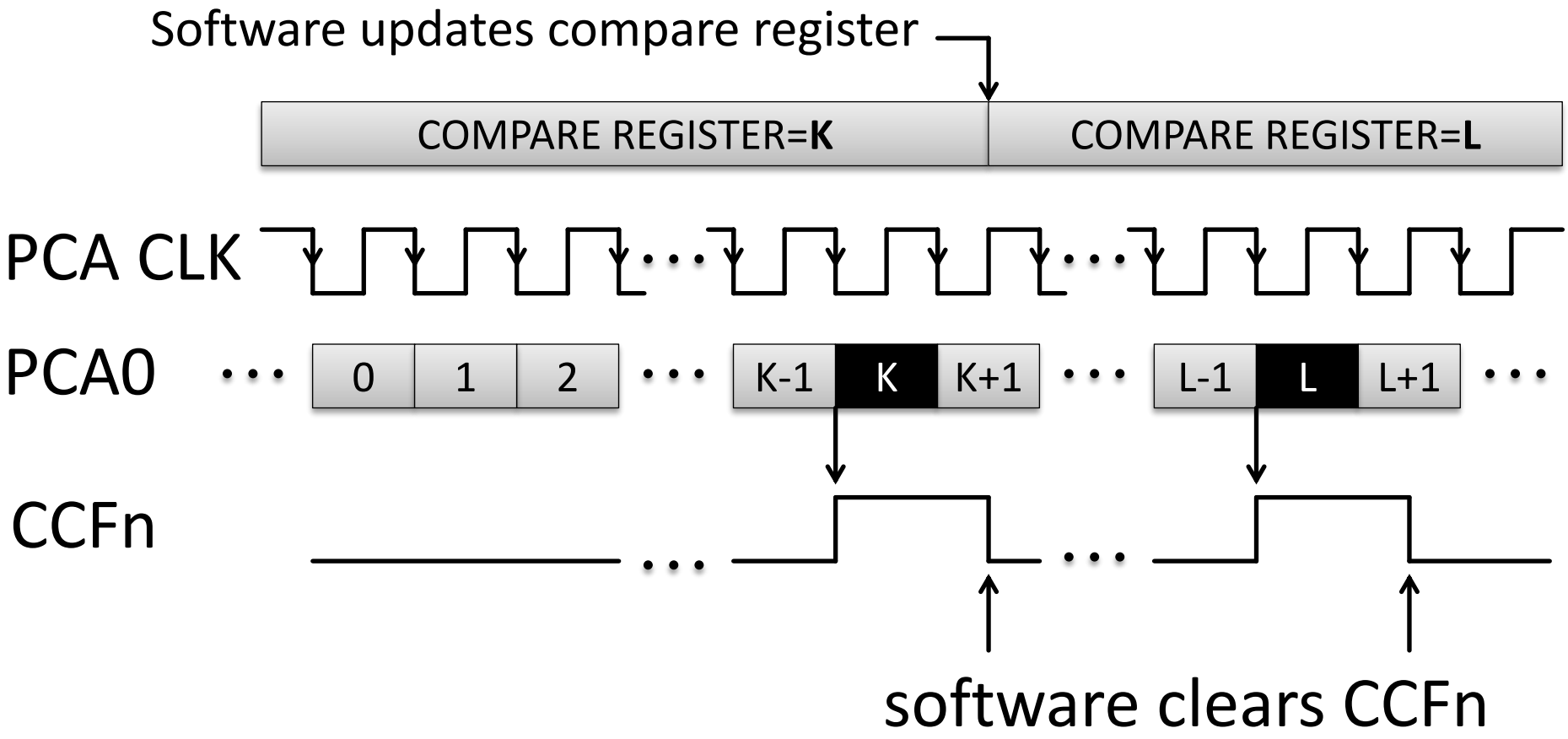


PCA ▸ Software timer mode



- **A CCFn 1-re vált, ha a PCA0 = PCA0CPn**
- Detektálás után CCFn törlése, új PCA0CPn érték beírása
- Változatos szoftveres időzítésekhez
- **FONTOS:** *Írási sorrend: PCA0CPLn, PCA0CPHn*

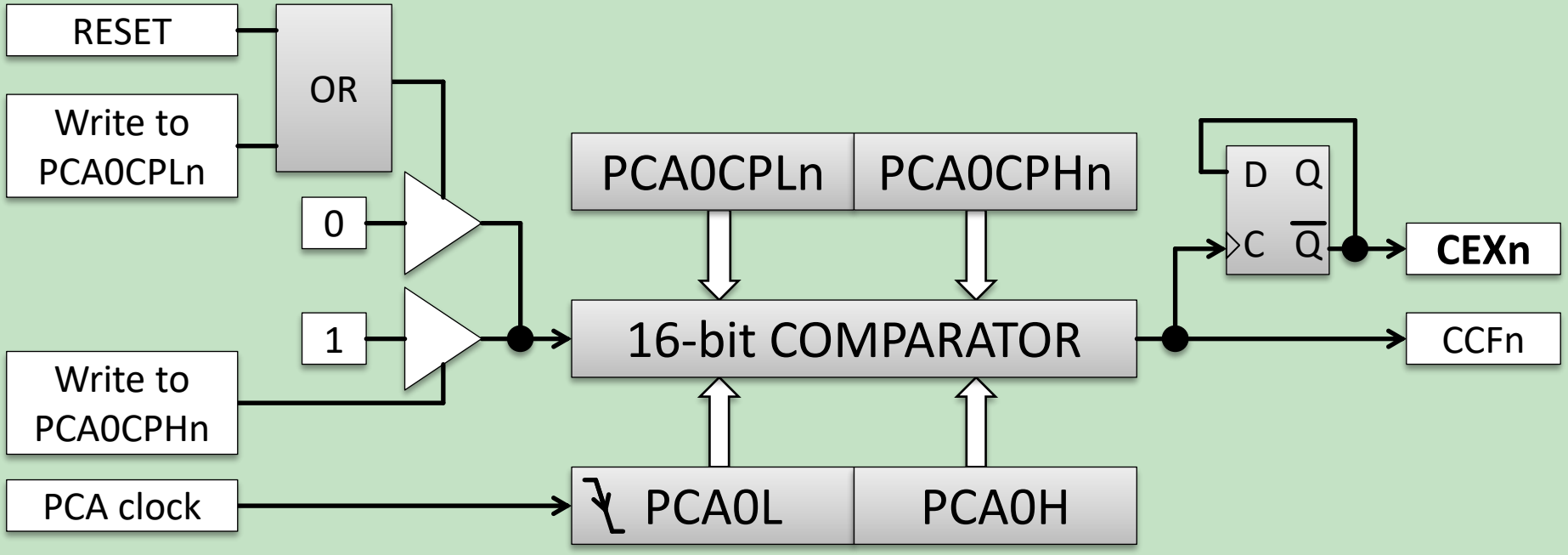
PCA ▸ Software timer mode



PCA ▸ Software timer ▸ Alkalmazások

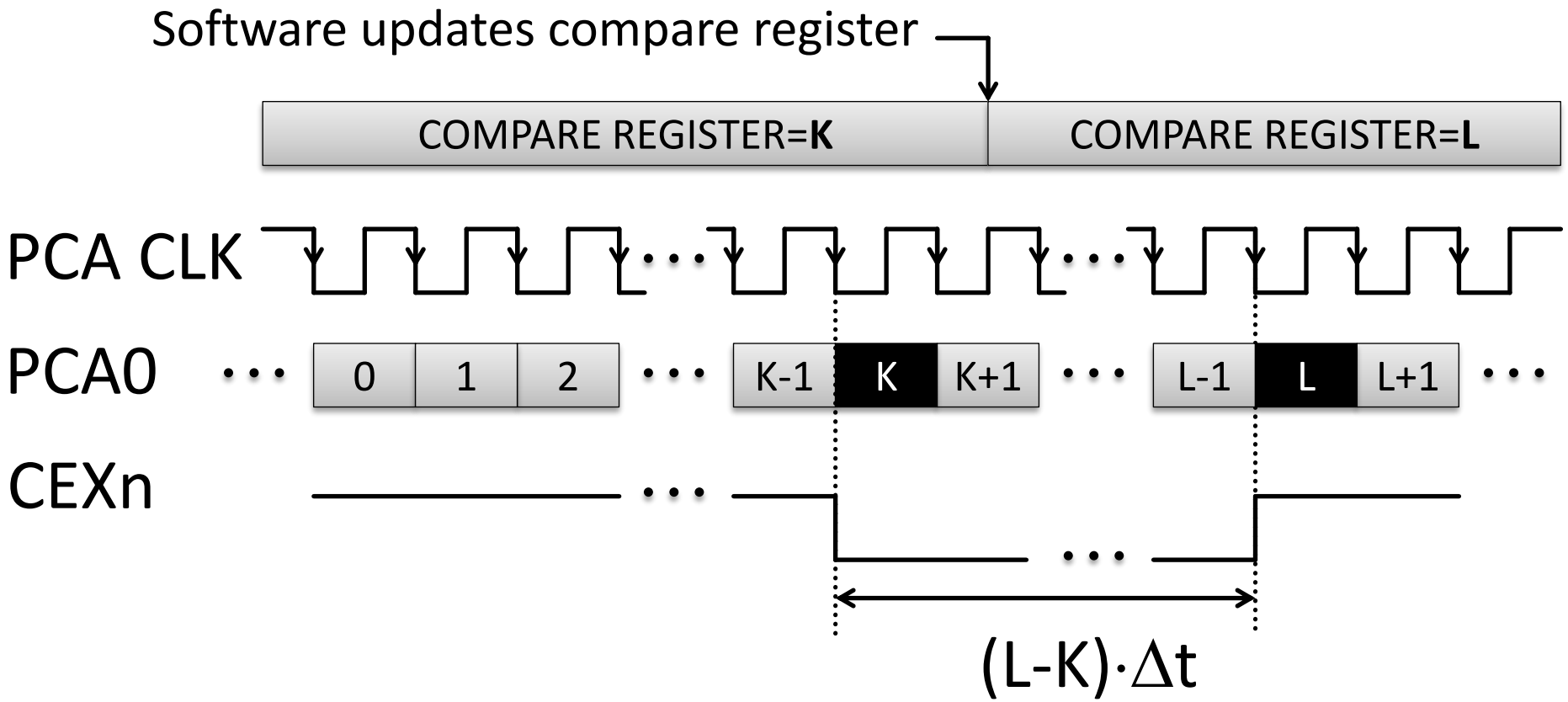
- Programozható késleltetés
- Adott idő múlva bekövetkező megszakítás
- Növekvő/csökkenő időnként bekövetkező megszakítás
- Forráskódok?

PCA ▸ High-speed output mode



- Azonos a software timer móddal, de kimenő jelet is ad
- A jel ellenkezőjére vált, ha a komparátor kimenete aktív
- Általános váltási idejű logikai jelek előállítására
- **FONTOS!** *Írási sorrend: PCA0CPLn, PCA0CPHn*

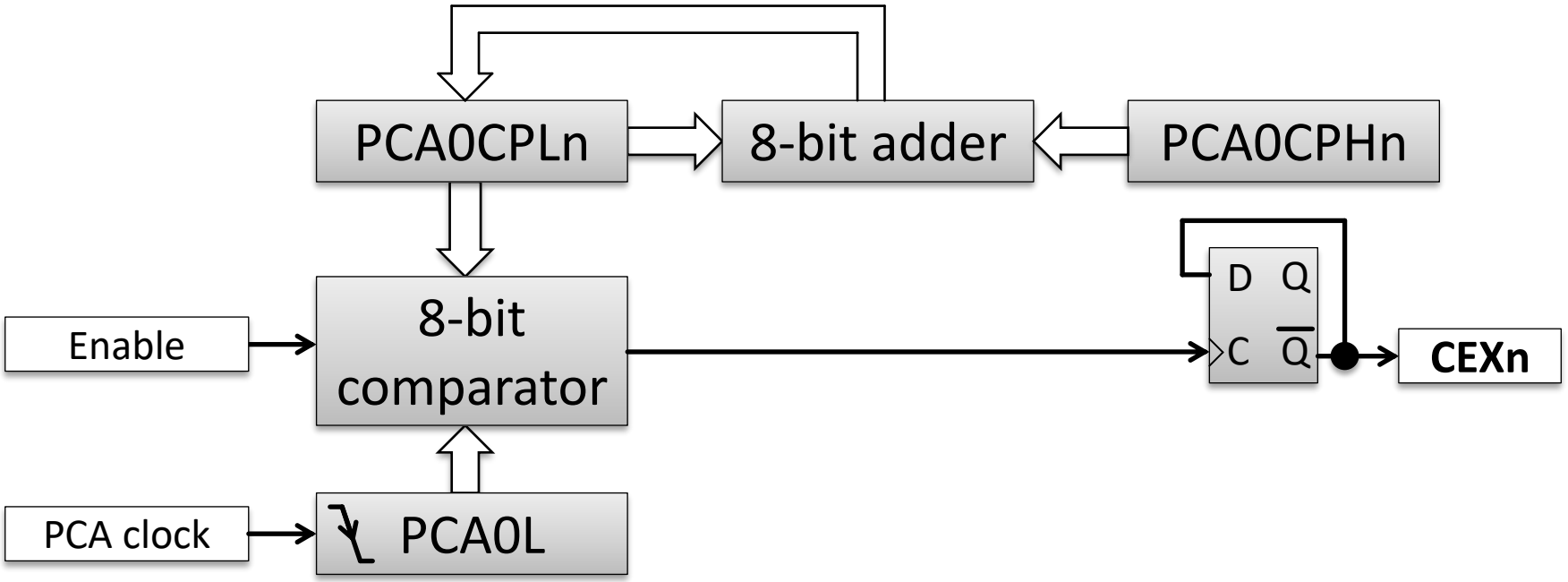
PCA ▸ High-speed output mode



PCA ▸ High-speed output ▸ Alkalmazások

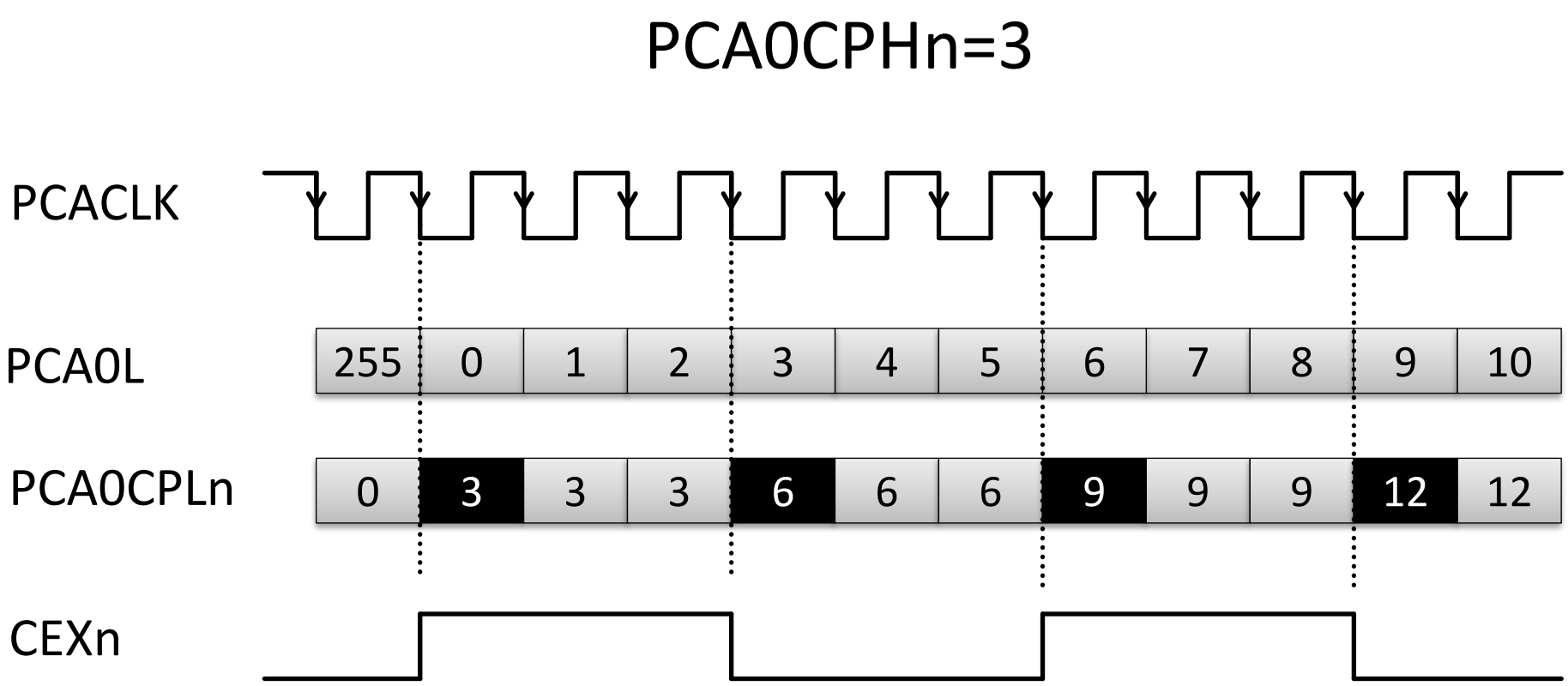
- Periodikus négyszögjel
- Növekvő/csökkenő időnként változó jel
- Változó szélességű impulzusok (PWM)
- Változó időközönkénti impulzusok
- Egymáshoz képest precízen időzített jelváltások
  - több modul használatával több jel
- A jelváltásokkor megszakítás
  - ez állíthatja be a következő jelváltási időt
  - részben szoftveres, befolyásolja a minimális időt!

PCA ▸ Frequency output mode

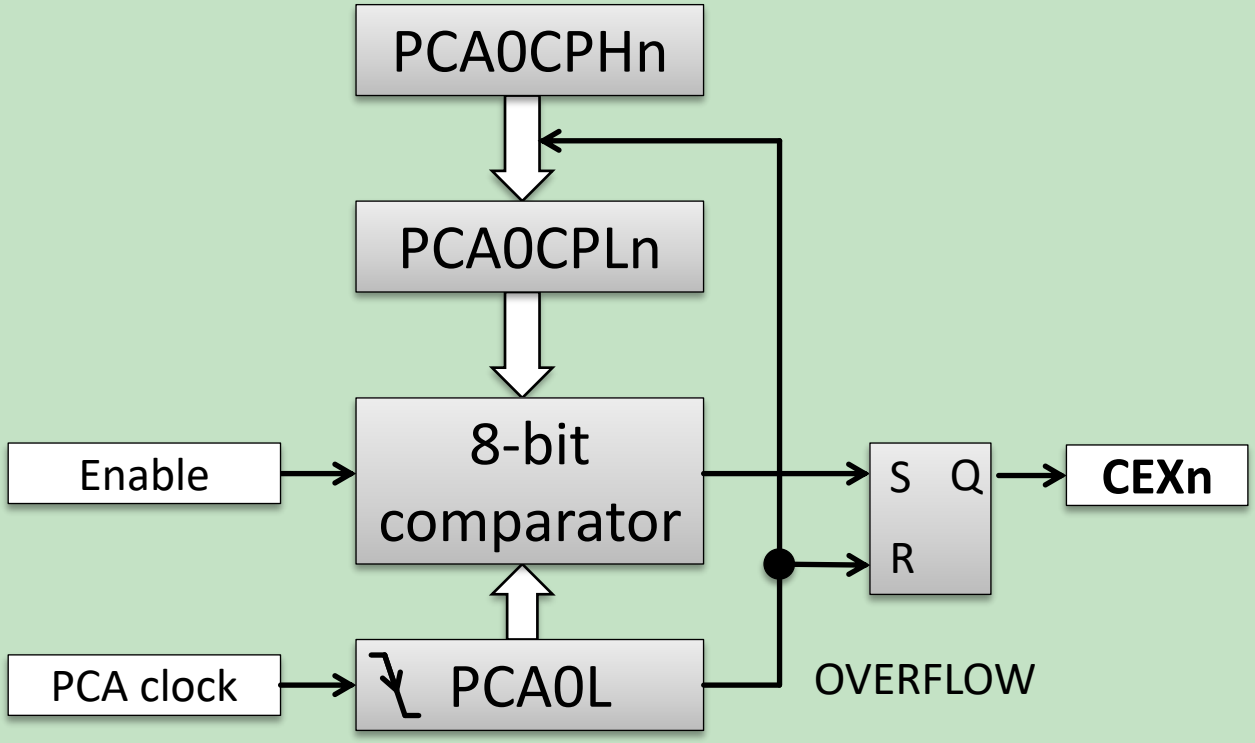


- 8-bit felbontású teljesen hardveres négyszögjel-forrás
- 50% kitöltési tényező
- $f=f_{PCA}/PCA0CPHn/2$

PCA ▸ Frequency output mode példa

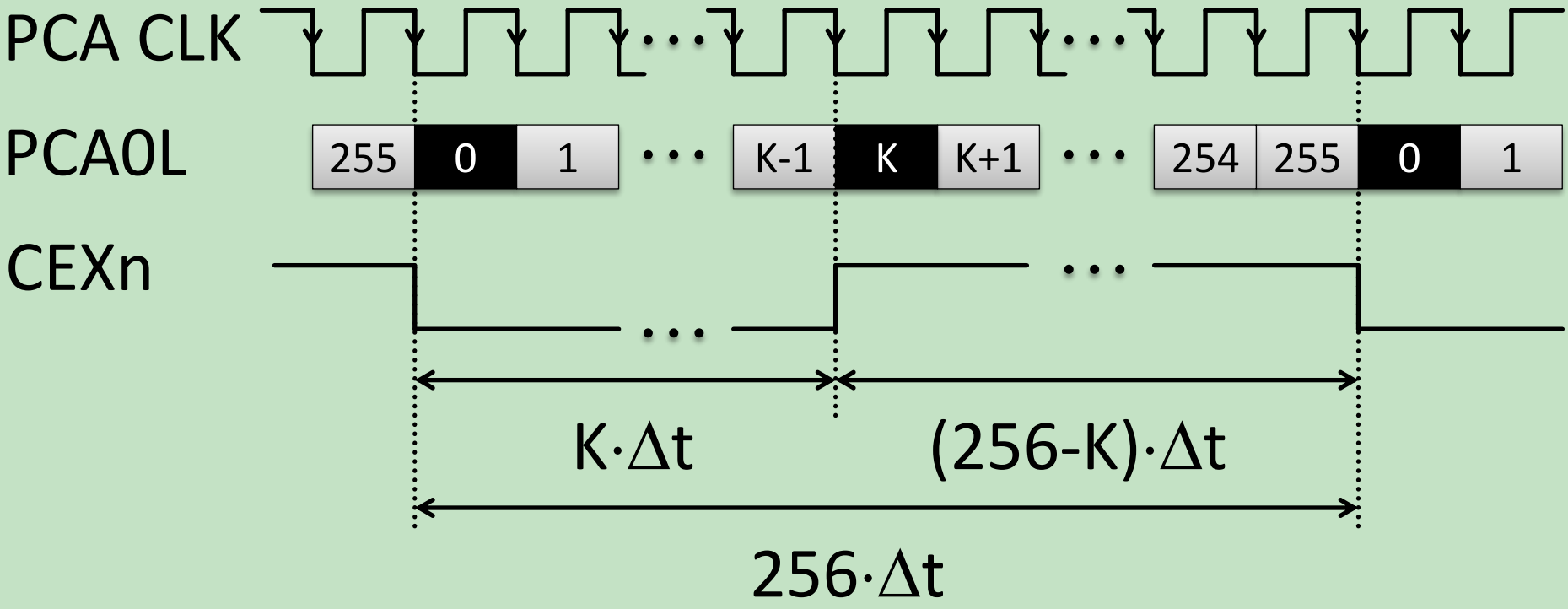


PCA ▸ Pulse width modulator (PWM) mode

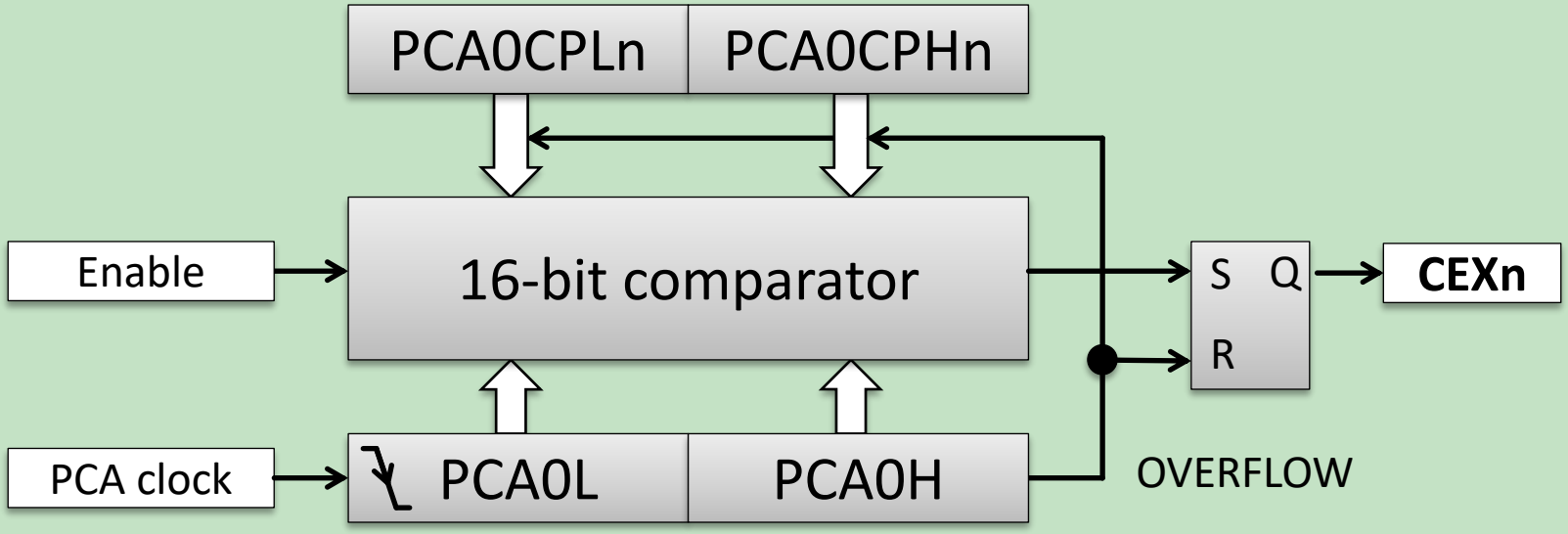


- 8-bites PWM, programozható kitöltési tényező
- $f_{PCA}/256$ frekvencia
- **FONTOS!** *Írási sorrend: PCA0CPLn, PCA0CPHn*

PCA ▸ Pulse width modulator (PWM) mode

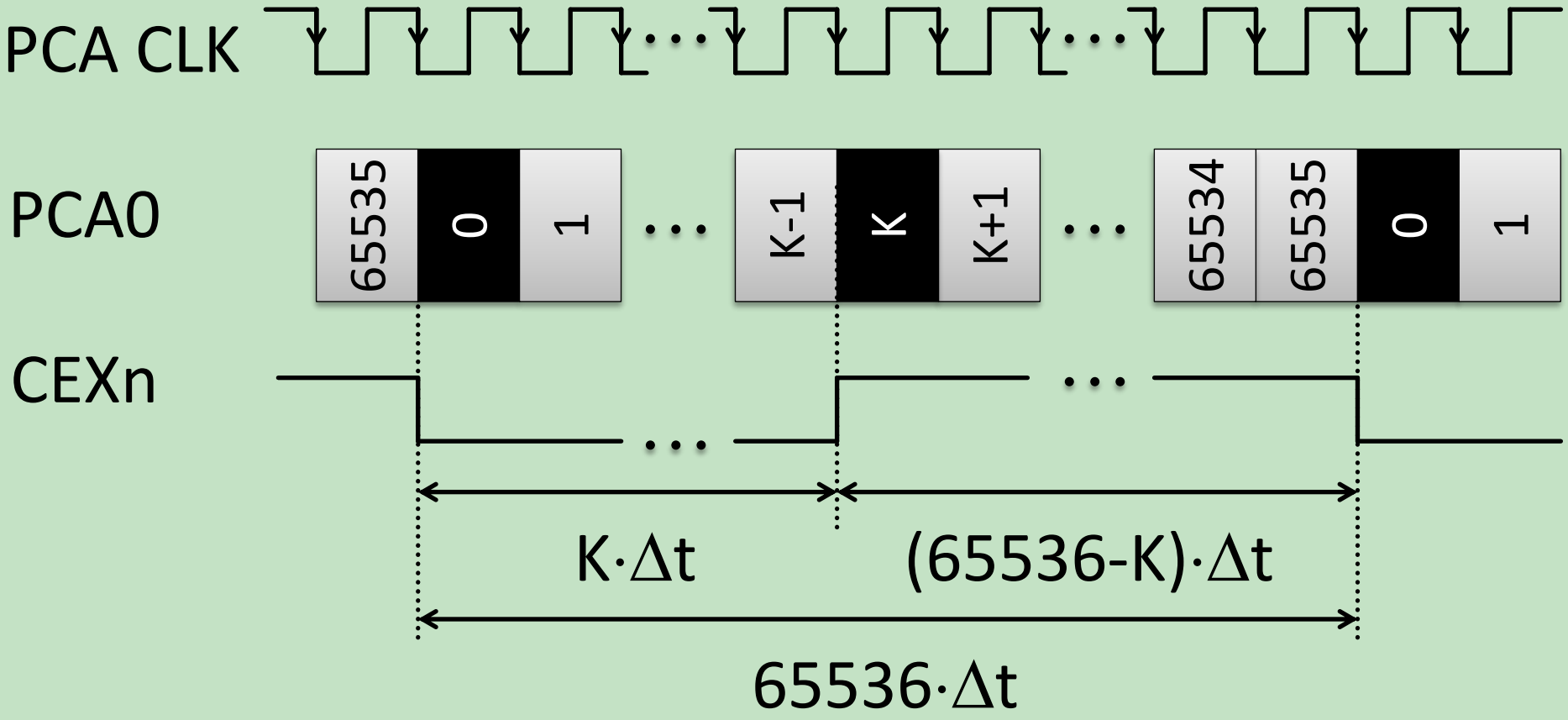


PCA ▸ Pulse width modulator (PWM) mode



- 16-bites PWM, programozható kitöltési tényező
- fpca/65536 a frekvencia
- **FONTOS!** *Írási sorrend: PCA0CPLn, PCA0CPHn*

PCA ▸ Pulse width modulator (PWM) mode

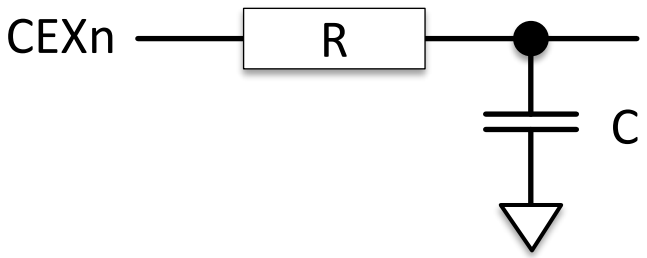


PCA ▸ PWM megfontolások

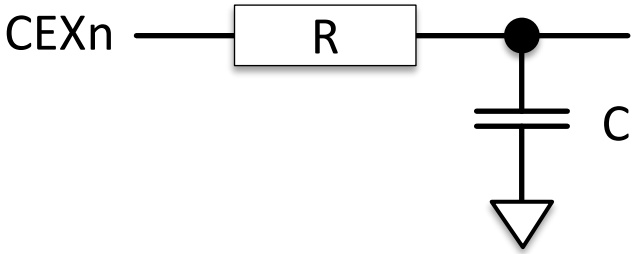
- A PWM periodikus jel
- Lassú rendszerek vezérlése
  - A periódusidőnél jóval hosszabb reagálási idő
  - A rendszer átlagolja (integrálja) a PWM jelet
- Példák
  - D/A konverzió
  - LED/lámpa fényerő
  - Motor fordulatszám
  - Fűtőszál
  - Peltier elem

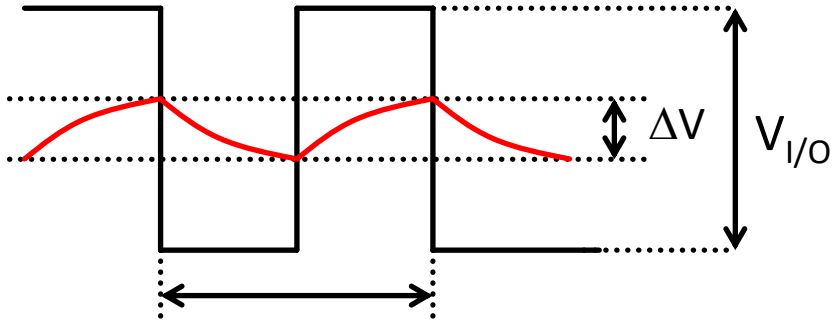
PCA ▸ PWM megfontolások

- Analóg jel?
- A frekvencia elég nagy a rendszer sebességéhez képest?
- Szűrés szükséges?
- Mekkora a hullámossága szűrés után?
- A logikai jel amplitúdója mekkora és milyen stabil?
- Ha mérjük a hatást, javítható a pontosság



PCA ▸ PWM alapú D/A konverzió



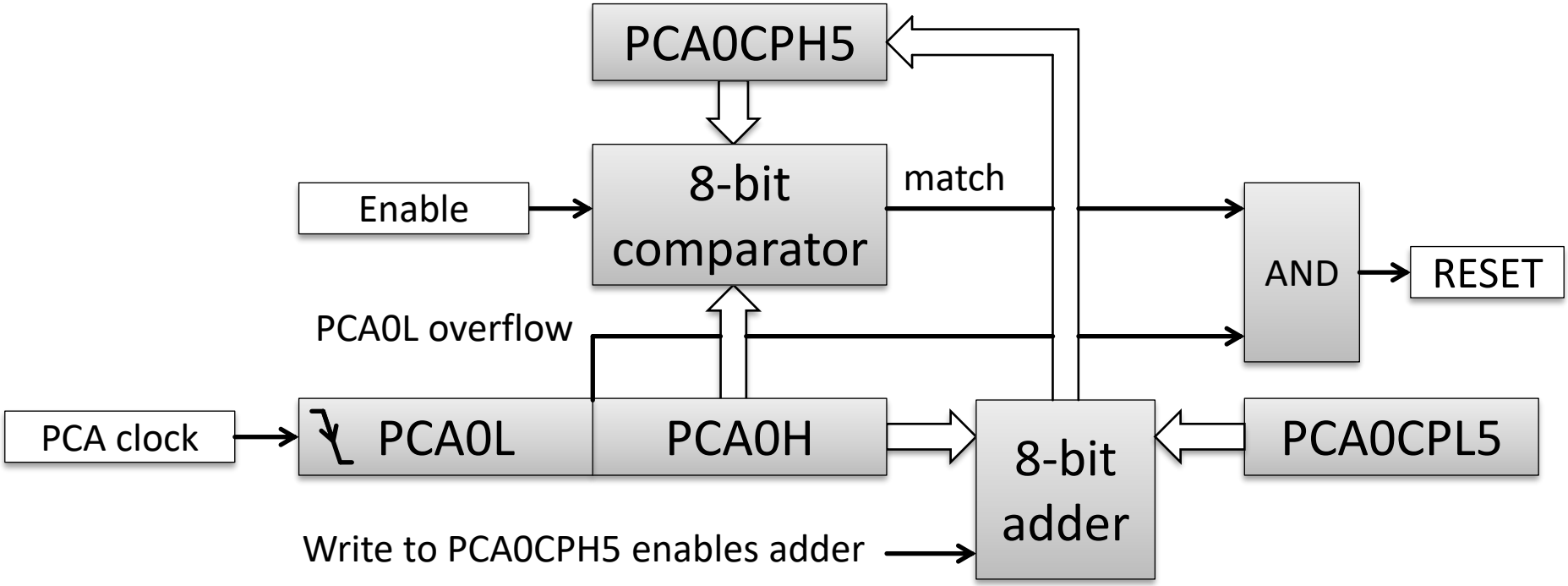


- **Közelítés**
- A legnagyobb ingadozás: 50% kitöltés
- T/2 ideig I=($V_{I/O}$/2)/R
- Töltés:I·T/2

$$\Delta V < \frac{\Delta Q}{C} = \frac{I \cdot \frac{T}{2}}{C} = \frac{\frac{V_{I/O}}{2}}{RC}\frac{T}{2}$$

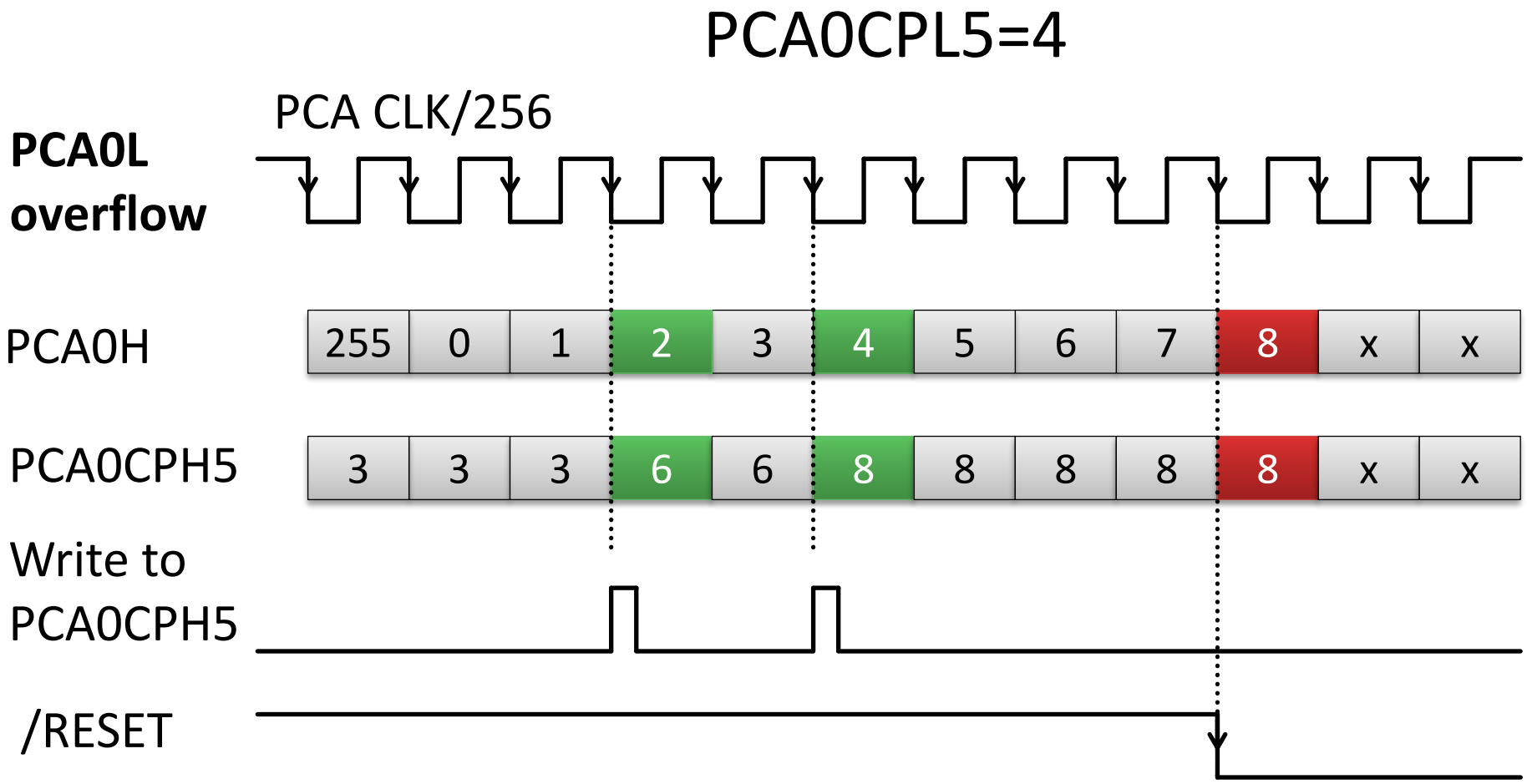
$$\frac{\Delta V}{V_{I/O}} < \frac{T}{4RC}$$

PCA ▸ Watchdog timer mode



- Bizonyos C8051Fxxx processzorokon
- Elfoglal egy PCA csatornát, ha aktív
- Ha engedélyezve van, zárolva van a PCA clock
- Ez alkalmazásokat korlátozhat

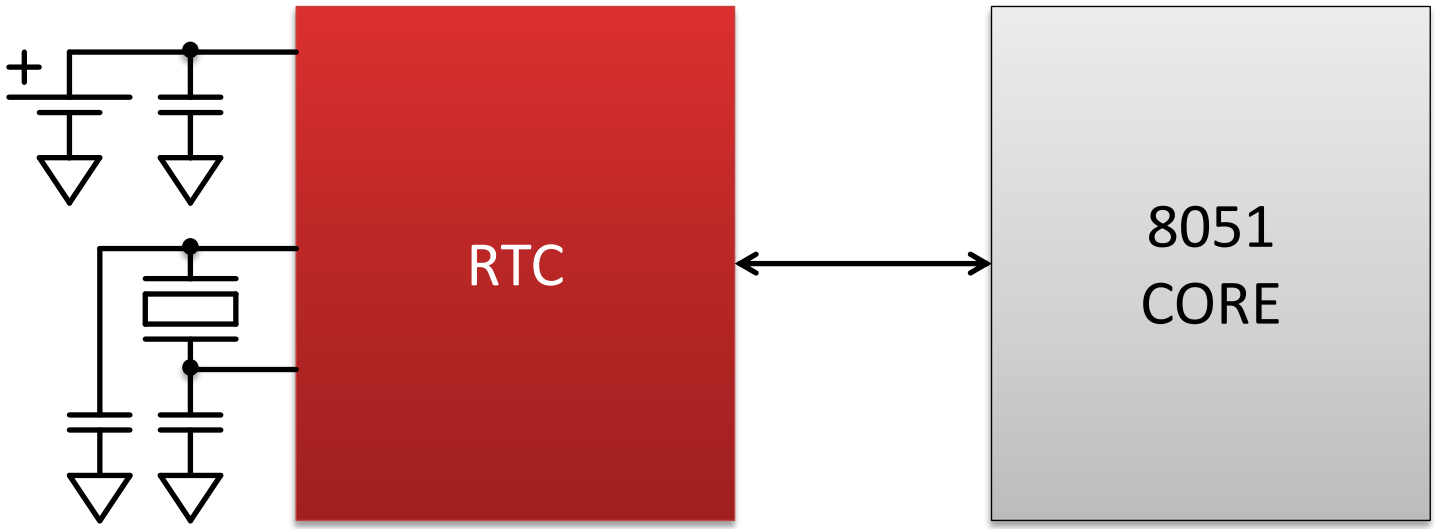
PCA ▸ Példák ▸ Watchdog timer



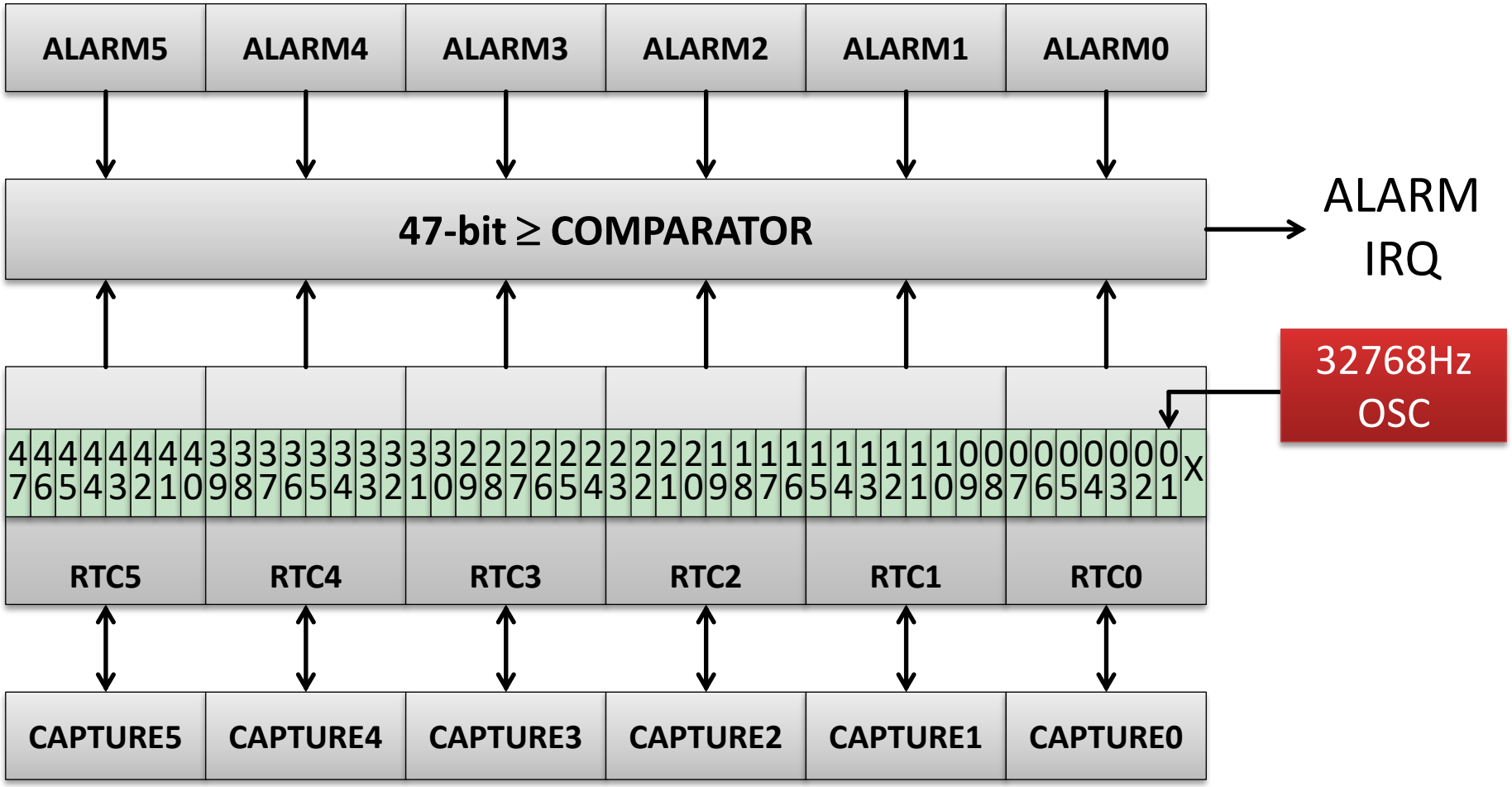
Valós idejű óra

RTC ▸ Real-time clock

- Valós idő mérésére, óra
- Elemről önállóan megy, ha a táp ki van kapcsolva
- Ébresztő funkció, felkelti a processzort
- Alacsony fogyasztás



RTC ▸ felépítés



RTC ▸ Alkalmazások

- Óra, naptár
  - másodpercek (4 byte : RTC5..RTC2, 2^{32} periódus)
  - másodperc konvertálása dátummá (1970.1.1. 0:00)
- Alarm
  - megszakítás
  - hosszú idejű altatások (nagyon alacsony fogyasztás)
- Processzor órajelét adhatja
  - alacsony fogyasztás
  - pontos
- PCA órajelét adhatja
  - RTC clock/8, (4096Hz, 65536 lépés: 16s)